ISO TC184/SC4/* WG10 N 62 (P____)

*Complete with EC (for Editing), PMAG, or WG

Date: 17th May 1996 **Supercedes SC4/ WG10 N 40 (P____)**

PRODUCT DATA REPRESENTATION AND EXCHANGE

Part: 13 **Title:** Description methods: architecture and development methodology reference manual

Purpose of this document as it relates to the target document is:

_X_ Primary Content
____ Issue Discussion
____ Alternate Proposal
____ Partial Content

Current Status: ISO WD

ABSTRACT:
This document specifies the architecture and methodology that is used for the development of ISO 10303.

KEYWORDS:

Document Status/Dates (dd/mm/yy)

Part Documents	Other SC4 Documents
_____ Released	_____ Released
_____ Project	_____ Working
_____ Working	_____ Editorial OK
_____ Technically Complete	_____ Technically
_____ Editorially Complete	Complete
_____ ISO Committee Draft	_____ Approved

Owner/Editor: Alison McKay
Address: Dept of Mechanical Engineering
The University of Leeds
Woodhouse Lane
Leeds LS2 9JT
United Kingdom
Telephone/FAX: + 44 113 233 2217
E-mail: mckay@leva.leeds.ac.uk

Alternate: Julian Fowler
Address: PDT Solutions,
29, The Downs,
Altrincham,
WA14 2QD,
United Kingdom
Telephone/FAX: + 44 161 976 6874
E-mail: jfowler@pdtsolutions.co.uk

Comments to Reader

This version of this document uses the latest available ISO.STY file. Issues relating to the ISO.STY file should not be addressed against this document.

© BSI 1995.
This document may be reproduced solely for the purpose of review/comment providing that all such copies include this copyright notice.

Contents

Figures

Tables

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liason with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

International Standard ISO 10303–13 was prepared by Technical Committee ISO 184, *Industrial automation systems and integration,* Subcommittee 4, *Industrial data and global manufacturing programming languages.*

This is the first edition of this part of ISO 10303.

Annexes A and B form an integral part of this part of ISO 10303. Annexes C,D,E,F,G are for information only.

This part is one in a series of parts which together form the International Standard ISO 10303, Industrial automation systems and integration – Product data representation and exchange. At the time of publication of this part of ISO 10303, the following parts had been registered for international ballot:

- Part 1 Overview and fundamental principles;
- Part 11 Description methods: The *EXPRESS* language reference manual;
- Part 21 Implementation methods: Clear text encoding of the exchange structure;
- Part 31 Conformance testing methodology and framework: General concepts;
- Part 32 Conformance testing methodology and framework: Requirements on testing laboratories and clients;
- Part 41 Integrated generic resources: Fundamentals of product description and support;
- Part 42 Integrated generic resources: Geometric and topological representation;
- Part 43 Integrated generic resources: Representation structures;
- Part 44 Integrated generic resources: Product structure configuration;
- Part 45 Integrated generic resources: Materials;
- Part 46 Integrated generic resources: Visual presentation;

- Part 47 Integrated generic resources: Shape variation tolerances;
- Part 49 Integrated generic resources: Process structure and properties;
- Part 101 Integrated application resources: Draughting;
- Part 104 Integrated application resources: Finite element analysis;
- Part 201 Application protocol: Explicit draughting;
- Part 202 Application protocol: Associative draughting;
- Part 203 Application protocol: Configuration controlled 3D design of mechanical parts and assemblies;
- Part 207 Application protocol: Sheet metal die planning and design.

The reader may obtain information on the other Parts of ISO 10303 from the ISO Central Secretariat.

The numbering of the parts of the International Standard reflects its structure:

- Parts 11 and 12 specify the description methods;
- Parts 21 to 26 specify the implementation methods;
- Parts 31 to 33 specify the conformance testing methodology and framework;
- Parts 41 to 49 specify the integrated generic resources;
- Parts 101 to 105 specify the integrated application resources;
- Parts 201 to 230 specify the application protocols;
- Parts 301 to 330 specify the abstract test suites;
- Parts 501 to 518 specify the application interpreted constructs.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product, independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and archiving.

ISO 10303 is organized as a series of parts, each published separately. The parts of this International Standard fall into one of the following series: description methods, integrated resources, application protocols, abstract test suites, implementation forms, and conformance testing. The series are described in ISO 10303–1. This part of ISO 10303 is a member of the description methods series.

The information generated about a product during its design, manufacture, use, maintenance, and disposal is used for many purposes during that life cycle. The use may involve many computer systems, including some that may be able to represent their product information in a common computer-interpretable form that is required to remain complete and consistent when exchanged among different computer systems.

This part of ISO 10303 documents the architecture of ISO 10303 (i.e. the components and the relationships between them) and the methods which support the development of ISO 10303.

The purposes of this part of ISO 10303 are:

- to provide the definitive statement of the current architecture and development methods;
- to present the fundamental assumptions and the fundamental concepts for the architecture and methodology.

It will thus:

- be a basis for the development and extension of ISO 10303;
- enable the wider application of ISO 10303 methods;
- be a basis for training in the methods.

The architecture and methods described in this part of ISO 10303 are specified in terms of processes, roles and involvements, rather than working groups, projects or committees. It focuses on application protocols which include the data models that are the parts of ISO 10303 whose implementation enables product data exchange. The focus of this part of ISO 10303 is a description of the methods in terms of their use in developing and enhancing the capabilities of ISO 10303 and other related standards. It is not an historical perspective.

The intent of this part of ISO 10303 is to make the architecture and development methodology available to a wide audience.

The target audiences for this document are those contributing to ISO 10303, including:

- ISO TC184/SC4 and its constituent national standard bodies and liaisons;

- ISO 10303 project leaders;
- experts in information technology, data modelling and systems integration;
- ISO 10303 developers.

EXAMPLE 1 – Those people who ensure integration between the different architectural elements of ISO 10303 and those people who specify application requirements are examples of ISO 10303 developers.

This part is divided into three sections. The first presents the architecture of ISO 10303, describing its different components and relationships. The second describes the methods used to develop key components: application protocols, integrated resources and application interpreted constructs. The third gives implementation principles that the ISO 10303 architecture and development methodology are designed to support.

Successful data exchange requires a contract between exchange partners which defines the form of the information to be exchanged. An exchange standard is a public contract which allows a wide range of partners to exchange data. The mechanism for measuring success is defined by the exchange partners.

Industrial automation systems and integration - Product data representation and exchange -

Part 13: Description methods: STEP architecture and development methodology

1 Scope

This part of ISO 10303 describes the architecture and methodology used in the development of ISO 10303. It also includes the basic assumptions and principles on which the architecture and development methodology are based.

The following are within the scope of this part of ISO 10303:

– activities preceding the registration of an ISO 10303 application protocol as an International Standard;

NOTE 1 – ISO 10303 application protocols define information structures for exchange in an application context and, as such, form the basis of any conforming exchange of product data.

– the architecture of ISO 10303;

– integration methods for developing the ISO 10303 integrated resources;

– methods for designing ISO 10303 conformance classes;

– methods for interpreting ISO 10303 integrated resources to produce ISO 10303 application protocols;

– implementation methods.

The following are outside the scope of this part of ISO 10303:

– procedures, practices and language usage guidelines for the development of ISO 10303;

– the methodology for the development of *EXPRESS*;

– conformance testing methodology and frameworks, including the development of abstract test methods;

NOTE 2 – These are described in the ISO 10303-30 series of parts.

– the methodology for the development of abstract test suites;

NOTE 3 – This is described in ISO 10303-33.

– system architectures and methods associated with implementations of product data exchange, shared product databases, or archiving.

NOTE 4 – See bibliography (Annex C) for more details of out of scope items.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO 10303. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 8824-1:1994 *Information technology — Open systems interconnection — Abstract Syntax Notation One (ASN.1) — Part 1: Specification of basic notation.*

ISO 10303-1:— *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:— *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: the EXPRESS language reference manual.*

ISO 10303-21:— *Industrial automation systems and integration — Product data representation and exchange — Part 21: Implementation methods: clear text encoding of the exchange structure.*

ISO 10303-31:— *Industrial automation systems and integration — Product data representation and exchange — Part 31: Conformance testing methodology and framework: general concepts.*

ISO 10303-32:— *Industrial automation systems and integration — Product data representation and exchange — Part 32: Conformance testing methodology and framework: requirements on testing laboratories and clients.*

ISO 10303-33:— *Industrial automation systems and integration — Product data representation and exchange — Part 33: Conformance testing methodology and framework: abstract test suites.*

ISO 10303-41:— *Industrial automation systems and integration — Product data representation and exchange — Part 41: Integrated generic resources: Fundamentals of product description and support.*

ISO 10303-43:— *Industrial automation systems and integration — Product data representation and exchange — Part 43: Integrated generic resources: Representation structures.*

3 Definitions

3.1 Terms defined in ISO 10303-1

This part of ISO 10303 makes use of the following terms defined in ISO 10303-1:

- abstract test suite;
- application;
- application activity model;
- application context;
- application interpreted model;
- application object;
- application protocol;
- application reference model;
- application resource;
- assembly;
- component;
- conformance class;
- conformance requirement;
- data;
- data exchange;
- data specification language;
- exchange structure;
- generic resource;

- implementation method;
- information;
- information model;
- integrated resource;
- interpretation;
- mapping table;
- presentation;
- product;
- product data;
- product information;
- product information model;
- resource construct;
- statement;
- structure;
- unit of functionality.

3.2 Terms defined in ISO 10303-11

This part of ISO 10303 makes use of the following terms defined in ISO 10303-11:

- data type;
- entity;
- entity data type;
- entity data type instance;
- instance;
- population;
- value.

3.3 Terms defined in ISO 10303-21

This part of ISO 10303 makes use of the following terms defined in ISO 10303-21:

- keyword;
- literal.

3.4 Terms defined in ISO 10303-31

This part of ISO 10303 makes use of the following terms defined in ISO 10303-31: [1)]

- conformance assessment process;
- conformance testing;
- conforming implementation;
- test purpose;
- verdict criteria.

3.5 Other definitions

For the purposes of this part of ISO 10303, the following definitions apply.

3.5.1 application assertion: a relationship between two application objects.

3.5.2 application interpreted construct: a logical grouping of interpreted constructs that is shared by two or more application interpreted models.

3.5.3 data specification: a formally defined model bounding a set of data.

EXAMPLE 2 – IDEF1X, NIAM, *EXPRESS-G* and *EXPRESS* are examples of data specification languages.

3.5.4 EXPRESS: the data specification language as defined by ISO 10303-11.

3.5.5 EXPRESS-G: the graphical data specification language as defined by ISO 10303-11.

3.5.6 EXPRESS-I: the instantiation language for a conceptual schema language as defined by ISO 10303-12.

4 Abbreviations

[1)] check once abstract test suite material is resolved

For the purposes of this part of ISO 10303, the following abbreviations apply.

ANSI/SPARC American National Standards Institute Standards Planning And Requirements Committee.

ICAM Integrated Computer-Aided Manufacturing

IDEF0 ICAM definition language zero

IDEF1X ICAM definition language one extended

ISOD ISO Directives Part 3

NIAM Nijssen's Information Analysis Method

5 Fundamental concepts and assumptions

5.1 ISO 10303

5.1.1 Introduction

ISO 10303 is based on the fundamental concepts and assumptions given in this clause.

5.1.2 Requirements

The following requirements have been identified:

a) the complete exchange of product data between application systems is required;

NOTE 1 – Completeness means that, from an application user's perspective, 100% of the required data is transferred between the applications with no diminishment of either its integrity or its semantics.

b) the complete archiving of such data is required;

NOTE 2 – Archiving means that, from an application user's perspective, data stored over any period of time remains accessible to, and usable by, applications.

NOTE 3 – Archiving may be based on defined data specifications and the use of conforming processors.

c) the sharing of product data between application systems is required;

NOTE 4 – The need for product data sharing is a requirement of many industrial enterprises. Each ISO 10303 application protocol is defined to support a specific application context that

is dependent upon the functionality and intended usage of the application systems that are to exchange data. The application context governs the coverage of the data specifications that define the information content of the exchange. An analogous context is required to govern the coverage of data specifications that define the information content of shared product data. However, this context is dependent upon the enterprise in which the product data sharing is to take place as well as the functionality and intended usage of the application systems that are to share data. A consequence of this additional enterprise dependency is that it is infeasible for ISO 10303 to provide data specifications that fully support the implementation of product data sharing. ISO 10303 application protocols provide for the application dependent aspects of product data sharing. It may be possible to use the ISO 10303 integrated resources and development methodology to provide for the extra, enterprise-specific, aspects of product data but this provision in not in the scope of ISO 10303.

d) ISO 10303 is required to be more reliable and efficient than existing standards;

EXAMPLE 3 – Existing standards include national and industry standards.

NOTE 5 – The efficiency of ISO 10303 is considered in terms of the computer resources required to carry out an exchange of product data and the people's time required to specify and use the standard.

e) ISO 10303 is required to enable the exchange of product data relating to any stage of the product life-cycle;

NOTE 6 – The scope of this assumption, and the technology to be used, are captured in the definition of ISO 10303 which interprets these requirements into the following objective [ISO 10303-1] and is reproduced in the introduction of each part of ISO 10303.

> ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life-cycle of a product, independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and archiving.

f) the distinction between application-independent and application-dependent concepts must be identifiable.

g) to minimise the cost and impact of change, particularly on data and implementations, upwards and downwards compatibility of implementations is necessary to enable archiving and black box use of processors by end-users;

h) industry requires compatibility with other standards;

NOTE 7 – Areas of relevance identified include graphics, mark-up languages, information modelling, and application specific standards and classification schemes.

i) testable implementations are required.

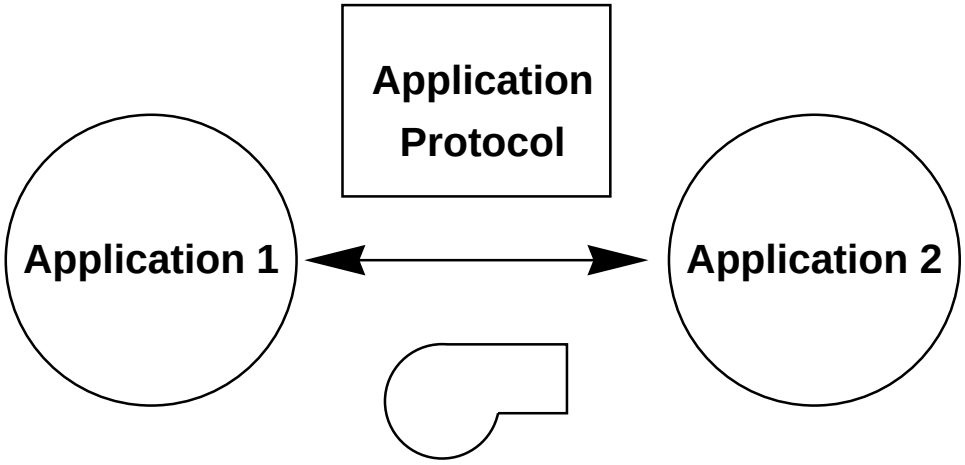

Figure 1 – Product data exchange

5.1.3 Fundamental assumptions

The following are fundamental assumptions:

a) ISO 10303 conforming processors are available to both exchange partners;

b) product data exchange does not require the implementation of all of the ISO 10303 parts[2)];

c) useful exchanges of product data only occur in an application context that is shared by the applications that are party to the interface.

5.1.4 Fundamental concepts

The following are fundamental concepts:

Product data exchange :

NOTE 1 – ISO 10303-1 defines data exchange as follows: "the storing, accessing, transferring and archiving of data."

NOTE 2 – Product data exchange is the transfer of product data between a pair of applications. An ISO 10303 application protocol defines the form of the product data that is transferred between a pair of applications. Figure 1 provides a schematic representation of these two statements. Each application operates upon its own copy of the product data and holds the product data in its own preferred form. The data conforming to the application protocol is transitory and defined only for the purposes of exchange.

Product data sharing : characterized by more than one application having access to and operating on a single copy of the same product data, potentially simultaneously.

NOTE 3 – In contrast to product data exchange, the applications do not hold the product data in their own preferred forms. ISO 10303 is designed to support product data exchange but it may be

[2)] Dave Sanford is to provide text for this – as agreed at Dallas

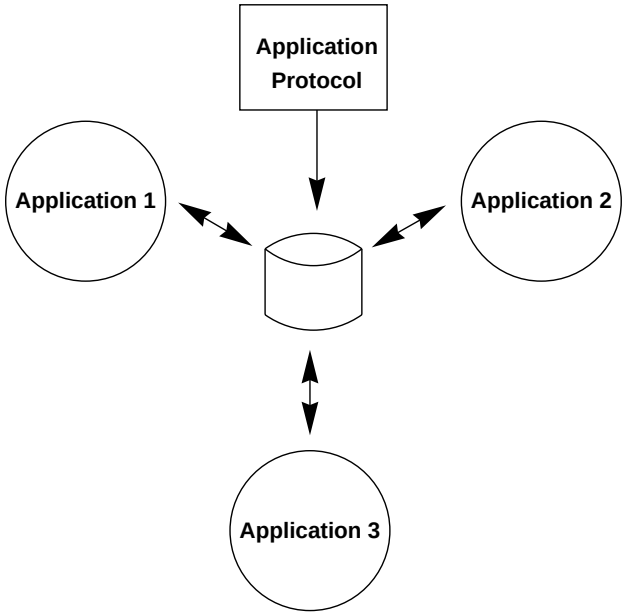

Figure 2 – Product data sharing using a single application protocol

used to support the development of interfaces between a single copy of the product data and the applications that share it. Figure 2 provides a graphical summary of these statements. The shared product data plays the role of an application in this scenario. ISO 10303 application protocols may be used to support the interfaces between the single copy of the product data and the applications that share it. Further, through the use of the ISO 10303 architecture and development methodology, the architectural elements of ISO 10303 may be used to support the realisation of the shared product data itself. In this case, the product data of prime interest is the integrated product data and not portions that are used by particular product data applications.

Product data archiving : the, usually long-term, storage of product data.

NOTE 4 – In the terminology of figure 1 and figure 2, the archive itself is an application. ISO 10303 application protocols may be used to support interfaces to the archive. Like product data sharing, the ISO 10303 architecture and development methodology may be used to support the development of the archived product data itself.
Archiving requires that the data conforming to an application protocol for exchange purposes is kept to be used at some other time. This subsequent usage may be through either product data exchange or product data sharing.

5.2 ISO 10303 architecture

5.2.1 Introduction

The ISO 10303 architecture is based on the fundamental concepts and assumptions given in this clause, derived from the fundamental concepts and assumptions for ISO 10303 in clause 5.1. It is a solution for the industrial requirements expressed in clause 5.1.2.

5.2.2 Fundamental assumptions

The following are fundamental assumptions:

a) the diverse nature of the applications between which product data exchange is to be supported requires that ISO 10303 be partitioned such that elements of the standard that support specific exchange requirements can be identified and used;

b) it is possible to use a single data structure to convey different meanings in different application contexts;

c) an ISO 10303 conforming product data exchange requires the implementation of an ISO 10303 application protocol;

NOTE 1 – This is an assumption about the way in which assumption b in clause 5.1.3 is met by the architecture.

NOTE 2 – Data exchange may be accomplished by conforming implementations of defined data specifications, namely the application interpreted models of application protocols or agreed subsets of application interpreted models of application protocols.

d) underlying consistency across data specifications contributes to their use in data specifications that are intended to support product data sharing;

e) expression of ISO 10303 data constructs in a computer-interpretable data definition language is necessary (but not sufficient) for unambiguous definition of data;

NOTE 3 – The languages or description methods are a fundamental part of the ISO 10303 development methods and *EXPRESS*, the ISO 10303 data specification language, is used by all other components of the architecture.

f) architectural accommodation for change and extensions over time while maintaining stability to encourage implementation and use is required;

5.2.3 Fundamental concepts

The following are fundamental concepts:

a) **product data representation**: ISO 10303 deals with the representation of product data independent of implementation methods.

NOTE 1 – The specification of the internal structure and operations of databases and application systems is outside the scope of ISO 10303.

b) **application context**;

c) **industrial information requirements**: application protocols are developed to satisfy clearly defined industrial information requirements;

NOTE 2 – Industrial information requirements are captured in the application activity model and the application reference model of an application protocol.

d) **reusable resources**: avoid the unnecessary duplication of data specifications in application protocols;

NOTE 3 – The integrated resources and application interpreted constructs are the reusable resources of ISO 10303.

e) **formal languages**: expression of ISO 10303 data constructs in a computer-interpretable data definition language is necessary (but not sufficient) for unambiguous definition of data.

NOTE 4 – The languages or description methods are a fundamental part of the ISO 10303 development methods: *EXPRESS*, the ISO 10303 data specification language, is used by all other components of the architecture.

5.3 ISO 10303 development methodology

5.3.1 Introduction

The ISO 10303 development methodology is based on the fundamental concepts and assumptions given in this clause, derived from the fundamental concepts and assumptions for ISO 10303 in clause 5.1. It supports the ISO 10303 architecture by defining methods for producing components of that architecture.

5.3.2 Fundamental assumptions

The following are fundamental assumptions about the ISO 10303 development methodology:

a) the development methodology is required to yield a standard that is interpretable by humans;

NOTE 1 – This requirement arises from the fact that the development of ISO 10303 depends heavily on human effort, human interaction and human acceptance now and in the future.

b) the development methodology is required to yield a standard that is interpretable by computers;

NOTE 2 – This requirement arises from the fact that ISO 10303 includes formal definitions which capture semantics and enable automated processing.

c) a single style and structure provides consistency of data and ease of understanding;

NOTE 3 – ISO 10303 abstracts common data semantics from multiple disciplines and phases of the product life-cycle into a single integrated structure.

d) traceability of the product-related data to the product to which it applies is required;

e) traceability of the exchanged data to the application context within which the exchange takes place is required;

f) application protocols that share an information requirement are required to use the same interpretation of the integrated resources;

g) extensibility is a prerequisite;

NOTE 4 – The following are extensible: the integrated resources, the series of application protocols, the framework for conformance testing and the series of description and implementation methods.

h) disciplined implementation of application protocols contributes to upward and downward compatibility;

i) the normative definition of an application protocol is developed to satisfy the documented information requirements of the application context.

5.3.3 Fundamental concepts

In order to support the architecture of ISO 10303, there are two fundamental concepts in the development methodology:

a) **integration**: the bringing together of like elements, information models – the result of the ISO 10303 integration process is a single information model;

NOTE 1 – A single information model is not necessarily a single *EXPRESS* schema.

b) **interpretation:** the bringing together of unlike elements, the information requirements of an application context and an information model – the result of the interpretation process is a single information model.

NOTE 2 – Interpretation includes relationship subsetting, role establishment, relationship constraint and attribute value constraint.

NOTE 3 – These are described in more detail in clauses 14.6 and 15.

Section I: The architecture of ISO 10303

6 Introduction

The ISO 10303 architecture and development methodology supports the development of product data exchange standards.

NOTE 1 – The exchange requirement is expressed in terms of interoperation between a pre-processor and a post-processor conforming to defined data. The availability of conforming processors facilitates interoperability between applications. This allows exchange partners (pre/post-processor users) to define, contractually, the information to be exchanged and how it is to be achieved.

The ISO 10303 architecture covers all elements of the standard, including:

a) the data architecture (application protocols and integrated resources);

b) description methods;

EXAMPLE 4 – *EXPRESS* (ISO 10303-11) and *EXPRESS-I* (ISO 10303-12) are examples of description methods.

c) implementation methods.

EXAMPLE 5 – Clear text encoding (ISO 10303-21) and the standard data access interface (ISO 10303-22) are examples of implementation methods.

The architectural aspects of ISO 10303 represented by the description and implementation methods are outside the scope of this part of ISO 10303.

This section serves the following purposes:

a) it provides an overview of the ISO 10303 architecture;

b) it describes each of the structural components of the ISO 10303 architecture;

c) it describes the functional aspects of the ISO 10303 architecture.

Figure 3 provides a high level summary of the elements of the ISO 10303 architecture and figure 4 shows the relationships between them.

NOTE 2 – The direction of the arrows in the diagram specifies dependencies. That is, the object at the tail of the arrow is dependent on the object at its head.

NOTE 3 – The architectural components shown in figure 3 are described in the following clauses of this document:

application context	clause 9.1
application activity model	clause 9.2
application reference model	clause 9.3
unit of functionality	clause 9.4
conformance class	clause 9.6

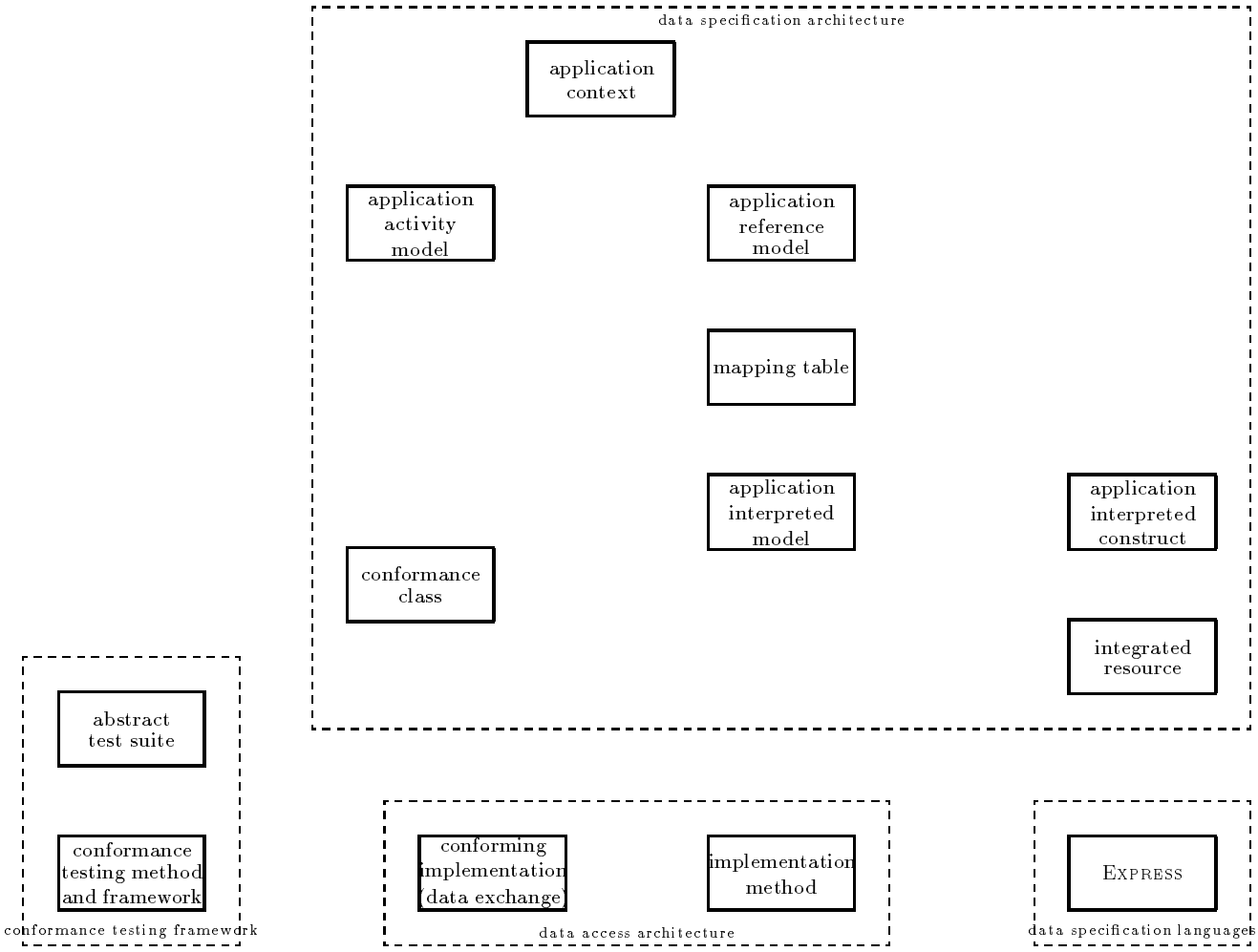

Figure 3 – The elements of the ISO 10303 architecture

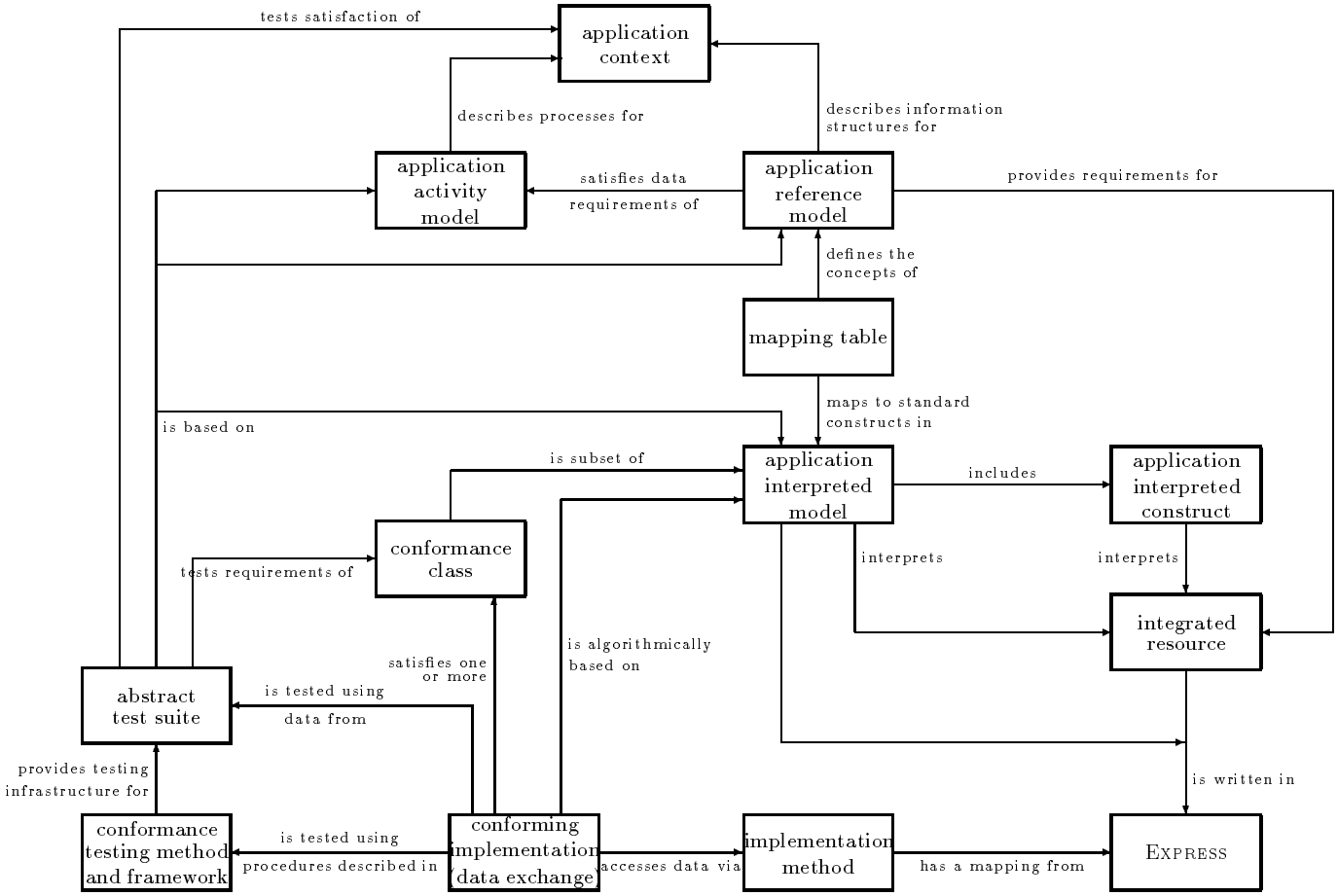

Figure 4 – The relationships between the elements of the ISO 10303 architecture

application interpreted model	clause 9.5
integrated resource	clause 10
application interpreted construct	clause 11
abstract test suite	clause 12

7 The architectural components of ISO 10303

7.1 Data specifications

The data specifications of ISO 10303 fall into the following categories:

a) **application protocols**: include (in the form of application interpreted models) data specifications that satisfy the specific product data needs of a given application context;

b) **application interpreted constructs**: data specifications that satisfy a specific product data need that arises in more than one application context;

c) **integrated resources**: data specifications (both generic and specific to a given application area) that support the consistent development of application protocols across many application contexts.

This distinction is reflected in the structure of ISO 10303, and its division into several series of parts.

NOTE 1 – The document architecture of ISO 10303 is given in annex F.

NOTE 2 – The data specification architecture of ISO 10303 is given in annex B.

Application protocols are the specification of an application context and the data constructs that are required by exchanges of product data in the application context;

NOTE 3 – The information structures of an application protocol include only those data constructs that are relevant to its application context.

NOTE 4 – Some application protocols define a logical classification of entities for their specified application context. These may be realised as conformance classes of the application protocol.

NOTE 5 – Application protocols include well defined usages of the integrated resources with conformance criteria, expressed as schemas reflecting the information requirements of different applications, on which implementations are based.

Integrated resources are the application-context-independent data constructs which constitute the minimal set of entities that are used by all application protocols;

NOTE 6 – The integrated resources include data constructs that are relevant to groups of closely related application contexts (integrated application resources) and data constructs that are applicable to product data in general (integrated generic resources).

NOTE 7 – The integrated resources reduce duplication (and size) and provide a consistent basis for application protocols.

NOTE 8 – Entities in the conceptual core fall into logical groups (such as shape and material) which are syntactically and structurally consistent. This enables them to be brought together to satisfy the information requirements of different applications. The generic product description resource and associated integrated resources capture this.

7.2 Description methods

Description methods are common mechanisms for the specification of the data constructs of ISO 10303.

NOTE 1 – The separation of data specifications (which use the description methods) from their implementation methods contributes to the upwards and downwards compatibility of implementations.

7.3 Implementation methods

Implementation methods are standard implementation techniques for the information structures specified in application protocols;

NOTE 1 – Each ISO 10303 implementation method defines the way in which data constructs specified using the ISO 10303 description methods are mapped to that implementation method.

NOTE 2 – An explicit framework for conformance and other types of testing is an integral part of the standard.

NOTE 3 – ISO 10303-31 defines several types of testing.

NOTE 4 – The standard includes criteria so conforming implementations are well defined. On these are based abstract test suites for use by users, implementation developers and independent test centres to enable the development of good processors and encourage expectations of trouble-free exchange.

7.4 Conformance testing

Abstract test suites are provide data which is used to test the conformance of a given implementation of ISO 10303.

NOTE 1 – An implementation of ISO 10303 for product data exchange is the combination of an application protocol and an implementation method.

8 The relationships between the architectural components of ISO 10303

As shown in figure 3, the data specification architecture of ISO 10303 is made up of different components. These components and their inter-relationships are described by this clause.

An *application activity model* describes the processes within and around an *application context* in terms of their activities and information flows. A corresponding *application reference model* describes the information requirements from the *application activity model* within the given

application context. An *application reference model* may contain *units of functionality*. Together, these four elements of the ISO 10303 architecture provide the information requirements which an *application interpreted model* is designed to support.

Every element of an *application interpreted model* relates to an information requirement in the corresponding *application reference model* . An *application interpreted model* may include *application interpreted constructs*. Both *application interpreted models* and *application interpreted constructs* are defined as interpretations of the *integrated resources*.

NOTE 1 – Interpretation is described in detail in clause 14.6.

Application interpreted models, *application interpreted constructs* and the *integrated resources* constitute the data specification aspects of the ISO 10303 architecture. All are documented using the *EXPRESS* data specification language.

NOTE 2 – The *EXPRESS* data specification language is described in ISO 10303-11.

A *conformance class* is a subset of an *application interpreted model*. A given *application interpreted model* may have several *conformance classes* but each *conformance class* is a subset of only one *application interpreted model*. Implicitly, *conformance classes* are documented using the *EXPRESS* data specification language.

Each ISO 10303 conforming implementation for data exchange is the combination of a particular application protocol with a single implementation method. The information structures of such an implementation are algorithmically based on the *application interpreted model* of the application protocol. Each implementation method contains a formally-defined syntax and a mapping from *EXPRESS* constructs onto the implementation method. This mapping is applied to the *EXPRESS* constructs in the *application interpreted model* and then used in conjunction with the syntax to specify the format of the information for a given *conformance class* of a given application protocol in the chosen implementation method. In this way, a single application protocol can be implemented in many ways.

NOTE 3 – The ISO 10303 implementation methods are described in the ISO 10303-20 series of parts.

ISO 10303 conforming implementation for data exchange are tested using the conformance testing methodology and framework.

NOTE 4 – The ISO 10303 conformance testing methodology and framework is described in the ISO 10303-30 series of parts.

Conformance testing uses abstract test suites that are based on the application interpreted model, application reference model and application activity model of an application protocol, and tests satisfaction with respect to the application context.

9 Application protocols

An application protocol is the specification of the product-related data constructs that conforming implementations support.

The ISO 10303 architecture is designed to support and facilitate the development of application protocols.

NOTE 1 – Application protocols are standardized in the ISO 10303-200 class of parts.

NOTE 2 – The development method for application protocols is given in clause 14.

An application protocol contains the following elements:

- scope;
- application activity model;
- application reference model;
- mapping table;
- application interpreted model;
- conformance class.

9.1 Scope

The scope of an application protocol describes the coverage of the information that is to be exchanged between applications in a given application context. An application context specifies the following:

- the functionality and/or technology of the application that is supported;
- the types of product/industry sector;
- the life-cycle stage;
- the discipline.

EXAMPLE 6 – The functionality of ISO 10303-203 is "configuration control", the types of product/industry sector are "piece-parts", the life-cycle stage is "design" and the discipline is "mechanical".

The scope of an application protocol fulfils the principle of traceability of data to industry needs.

NOTE 1 – The development method for application protocols is given in clause 14.1.

NOTE 2 – The scope of an ISO 10303 application protocol is written in natural language.

9.2 Application activity model

An application activity model is a specification of the general enterprise activities, and the information flows between them, that are relevant to the application context that is identified in the scope of the application protocol.

The application activity model describes the input and output information requirements of the activities within the application context. It identifies activities of an application area and the

information flows that are inputs or controls to the activities or that are outputs of the activities. It supports the analysis of the activities and information flows within the scope of the industry application. Further detailed analysis and design of data specifications within ISO 10303 is linked back to the in-scope activities and information flows.

NOTE 1 – The role of the application activity model is to capture the activities ("what is done") within an industry application, not the detailed processes ("how it is done"), that are likely to vary between organisations, or with time as the result of continuous improvement or business process re-engineering activities.

NOTE 2 – The development method for application activity models is given in clause 14.2.

NOTE 3 – The application activity model of an ISO 10303 application protocol may be developed and documented using the IDEF0 activity modelling methodology [2]. In accordance with this methodology, the application activity model includes diagrams that provide a graphical representation of the activities and a glossary that defines all activities and information elements in the model.

Application protocols that deal with different technical aspects of the same product or different life-cycle phases of the product may be related through their application activity models. Where an output of an activity that is in the scope of one application protocol serves as an input or control to activities within the scope of another application protocol, a point of overlap in the elements of the information flow exists and that is reflected in the application interpreted models of the application protocols.

NOTE 4 – These points of overlap in the elements of the information flows may not be identified explicitly in the individual application protocols.

9.3 Application reference model

An application reference model is a description of the product data that exists from the perspective of an identified application context. It captures the information requirements of the application protocol.

The application reference model results from a detailed analysis of the requirements of the application context. It includes the definition of application objects and assertions. The application reference model has minimal influence from data modelling technologies. It expresses as closely as possible the product data as defined by application area experts. The application reference model is prepared by application area experts.

These requirements are therefore described using the terminology of the application context, and form the basis not only for further development, but also for review and validation of the application protocol. Each application information requirement that is within the scope of the application activity model is expressed in the application reference model. Conversely, each element of the application reference model satisfies a documented information need of the application context. The application reference model describes fully the data needs of the application, using the terminology of the application.

NOTE 1 – The development method for application reference models is given in clause 14.3.

NOTE 2 – Development practices for the application reference models of ISO 10303 application protocols are given in [13].

NOTE 3 – The normative specifications of the application objects and assertions in the application reference model of an ISO 10303 application protocol are provided as natural language descriptions. The informative specification (of an information model) uses one of the following: NIAM, IDEF1X, or *EXPRESS*. The informative specification is given solely to aid understanding of the normative specifications.

NOTE 4 – As shown in figure 3, the application reference model of an application protocol depends upon the corresponding application activity model: it is a detailed description of the data that supports the activities and flows between the activities described in the application activity model.

9.4 Unit of functionality

A unit of functionality is an aggregate of application objects and assertions which is important in a given application context. It states the information requirements, in the form of application objects and assertions, that constitute one or more concepts within the context of an application reference model.

A unit of functionality is a mechanism for modularising the scope of an application protocol. A unit of functionality usually supports an application function or process. Units of functionality are used to organise and summarise the functionality of the application reference model.

EXAMPLE 7 – If a geometric modelling application has a requirement for wireframe geometry, then a unit of functionality may be defined which provides a grouping of those application objects in the application reference model which are intended to support geometric modelling using wireframe geometry.

NOTE 1 – The development method for units of functionality is given in clause 14.4.

NOTE 2 – The units of functionality of an ISO 10303 application protocol are written in natural language in the same terms as the application reference model.

An application reference model may contain one or more units of functionality.

9.4.1 Mapping table

A mapping table is a specification of the transformation from elements of the application reference model (application objects and assertions) to the elements of an application interpreted model through derivation from the integrated resources.

The interpretation process requires that the constructs in the application reference model are mapped to suitable constructs in the integrated resources. This process results in two sections of the application protocol: the application interpreted model and the mapping table. The mapping table includes the following:

– reference paths through the structure defined by the integrated resources;

– constraints on the reference paths, and data constructs therein, that are dictated by the application reference model.

The mapping table documents the relationship between data constructs from the integrated resources and application objects from the application reference model through the definition of reference paths. Further, it provides information about these reference paths.

EXAMPLE 8 – The mapping rules required to create, import, or interpret application objects into an application system are examples of the additional information provided by a mapping table.

During the interpretation of the integrated resources, the mapping of the correspondence between the application objects and the constructs of the application interpreted model are documented and maintained. The resulting mapping table shows the application interpreted model construct(s) required for each application object.

NOTE 1 – The development method for mapping tables is given in clause 14.5.

NOTE 2 – The reference paths and constraints of the mapping table of an ISO 10303 application protocol is specified in a syntax that has been designed specifically for mapping tables.

An application system that can read or write a data file based on an application interpreted model is necessary, though not sufficient, for using an application protocol. The mapping table provides the additional rules for transforming data defined in terms of the application interpreted model into the application objects that are defined in the application reference model.

9.5 Application interpreted model

An application interpreted model is a specification of elements of the integrated resources that have been interpreted to satisfy the information requirements of the application reference model. The application interpreted model specifies the implementable constructs of the application protocol.

The application interpreted model contains the data structure and its semantics used for the exchange of product data for a given application protocol. The data structure and product data semantics are taken from the integrated resources and interpreted to accommodate the precise semantics of the application reference model in terms of the integrated resource constructs.

NOTE 1 – The multiple uses of standard integrated resource constructs to meet a wide range of application requirements result in a high degree of consistency across the application interpreted models of separate application protocols. This gives the potential for the reuse of interface software and the sharing of common data between separate application contexts.

It is possible to develop different kinds of implementation of a given application interpreted model because application interpreted models are conceptual in nature.

NOTE 2 – Conceptual data models exclude any data constructs that are specific to a given implementation method.

EXAMPLE 9 – Primary and foreign keys are examples of implementation-specific data constructs for relational database implementations.

EXAMPLE 10 – A given application interpreted model may be used as the basis of either a file-based implementation in conjunction with ISO 10303-21 or a data access implementation based on ISO 10303-22.

Areas of overlap in application protocols may be indicated by commonality in the scope statements or the use of the units of functionality in their application reference models. These areas may also be indicated by the common use of integrated resource constructs within the application interpreted models. When two application protocols contain equivalent information requirements, these application protocols use the same interpretation of the integrated resource-constructs. Such interpretations may become candidate application interpreted constructs.

NOTE 3 – The development method for application interpreted models is given in clause 14.6.

NOTE 4 – Development practices for the application interpreted models of ISO 10303 application protocols are given in [12].

NOTE 5 – The application interpreted model of an ISO 10303 application protocol is documented in the following formats:

- application interpreted model *EXPRESS-G*;
- application interpreted model *EXPRESS* short listing;
- application interpreted model *EXPRESS* annotated listing.

The short listing of an application interpreted model includes references to the integrated resources and application interpreted constructs that are used in the application protocol and it contains the *EXPRESS* types, entity specialisations, rules, and functions that are specific to the application protocol.

The annotated listing of an application interpreted model is a self-contained *EXPRESS* schema. It is the result of explicitly including all constructs that are referenced, either directly or indirectly, from the short listing of the application interpreted model.

9.6 Conformance class

A conformance class is a definition of a subset of the application interpreted model that may be used as the basis for implementation and testing. Each subset defines a conforming implementation based on the application interpreted model; implementations based on any other subsets are not considered to be conforming.

An application interpreted model, as described above, provides the normative specification for data to be exchanged between product data applications. This provides the scope for implementations of product data exchange that conform to ISO 10303, and also the scope and boundaries for testing implementations. In order both to meet the needs of different computer systems used within a given industrial application and to maintain consistency of implementation and testing, two or more conformance classes may be defined for an application interpreted model.

NOTE 1 – If no conformance classes are defined for an application interpreted model then it is required that conforming implementations implement the complete application interpreted model. In this case, the full application interpreted model is the sole conformance class.

NOTE 2 – The development method for conformance classs is given in clause 14.7.

NOTE 3 – The conformance classes of an ISO 10303 application protocol are written in natural language and *EXPRESS*.

10 Integrated resources

Integrated resources are application-context-independent product data specifications that constitute the standard foundation that is used in the creation of ISO 10303 application interpreted

models.

They are data models that reflect and support the common requirements of many different product data application areas. The integrated resources constitute a single, conceptual model for product data. The constructs within the integrated resources are the basic semantic elements that are used for the description of any product at any stage of the product life-cycle. Products are characterised by their properties. Different groups of properties are used to characterize the same product at different stages of its life-cycle.

EXAMPLE 11 – A product may be characterized by its designed shape (which includes toleranced dimensions) and required material properties immediately after the design process has been completed. On the other hand, the same product may be characterized by its manufactured shape (which includes measured dimensions) and its actual material properties after it has been manufactured.

Thus, the properties that characterise a product do so indirectly, through their characterisation at different stages of the product life-cycle. These properties in turn may be represented in many different ways depending upon the nature of the stage of the life-cycle in which the product is being described.

EXAMPLE 12 – The shape property of a product may be represented using a boundary representation or a constructive solid geometry representation.

Further, each property representation may be presented in many different ways.

EXAMPLE 13 – The constructive solid geometry representation of a shape may be presented as a shaded image or a wireframe image.

NOTE 1 – More detailed descriptions of the modules which determine the overall structure of the integrated resources are given in annex E and ISO 10303-41.

Although the integrated resources are used as the basis for developing application interpreted models, they are not themselves intended for direct implementation: they define reusable components that are intended to be combined and refined to meet a specific need.

This classification of product data is the basis for all ISO 10303 data specifications. It is the framework upon which all of the integrated resources are built, and it is reflected in the application interpreted models of all application protocols.

NOTE 2 – The data specification architecture of ISO 10303 is given in annex B.

NOTE 3 – A description of the models that capture this framework is given in annex D.

NOTE 4 – Integrated resources are standardized in the ISO 10303-40 series class of parts.

NOTE 5 – The development method for integrated resources is given in clause 15.

NOTE 6 – The integrated resources of ISO 10303 are documented using the *EXPRESS* data specification language.

11 Application interpreted constructs

An application interpreted construct is the specification of the data structures and semantics that are used for the exchange of product data common to two or more application protocols.

Application protocols with similar information requirements are compared semantically to determine functional equivalence that, if present, leads to the use of common application interpreted constructs.

NOTE 1 – These information requirements may be in the form of units of functionality.

When two or more application protocols have common information requirements, the same interpretation of the integrated resources is used in the application interpreted models of each application protocol. This interpretation is accomplished by the inclusion of a common module, an application interpreted construct, in each of the application protocols. An application interpreted construct identifies the potential for shared data between industry applications. All application protocols which have been identified as sharing an information requirement use the same application interpreted construct in its entirety.

The main purpose of an application interpreted construct is to provide a mechanism to identify and encapsulate the common requirements of distinct application contexts represented by their respective application reference models.

A necessary, but not sufficient, condition for interoperability between processors that share common information requirements is common data models. Application interpreted constructs provide this capability.

Equivalences between the functional requirements of different application protocols, and so the need for application interpreted constructs, are identified during the interpretation process.

NOTE 2 – Application interpreted constructs are standardized in the Parts of ISO 10303 which start at ISO 10303-501.

NOTE 3 – The development method for application interpreted constructs is given in clause 16.

NOTE 4 – The application interpreted constructs of ISO 10303 are documented in *EXPRESS*.

12 Abstract test suites

An abstract test suite is the specification, in a non-specific or parametrised form, of the collection of abstract test cases from which executable test cases may be derived and used in assessing the conformance of an implementation to the data specification contained in an application interpreted model and the other elements of the ISO 10303 architecture upon which an application interpreted model depends.

The importance of testing and testability within ISO 10303 is reflected by a standardised framework and methodology for conformance testing.

NOTE 1 – Experience in other areas, such as the OSI standards for Open Systems, has shown that standardisation of abstract test suites is an essential prerequisite to repeatability and consistency of testing, and therefore of mutual recognition of test results across regional or national boundaries.

NOTE 2 – Abstract test suites are standardized in the ISO 10303-300 class of parts .

NOTE 3 – The development method for abstract test suites is given in clause 17.

NOTE 4 – The abstract test suite of an ISO 10303 application protocol is written in either *EXPRESS-I*or the ISO 10303-21 physical file format.

Section II: The ISO 10303 development methodology

13 Introduction

The ISO 10303 development methodology supports the development of application protocols and the resources that application protocols require. It covers all elements of the standard, including:

a) application protocols, integrated resources and application interpreted constructs;

b) description methods;

EXAMPLE 14 – *EXPRESS* (ISO 10303-11) and *EXPRESS-I* (ISO 10303-12) are examples of description methods.

c) implementation methods.

EXAMPLE 15 – Clear text encoding (ISO 10303-21) and the standard data access interface (ISO 10303-22) are examples of implementation methods.

NOTE 1 – The methodologies for the development of ISO 10303 description methods and implementation methods are outside the scope of this part.

This section describes the methods for the development of application protocols, integrated resources and application interpreted constructs.

14 Application protocols

This clause describes the method of developing an application protocol. It assumes that the necessary integrated resources and application interpreted constructs are developed once a need is identified during the application protocol development.

NOTE 1 – The components of an application protocol are described in clause 9.

The scope, application activity model, and application reference model of an application protocol are defined before the other components.

NOTE 2 – The process of defining the scope, application activity model, and application reference model of an application protocol is usually iterative in nature.

This definition, sample product data and usage scenarios provide the foundation for the development of an application protocol.

NOTE 3 – The usage scenarios may be used in the subsequent validation of the application reference model and the application interpreted model.

NOTE 4 – The development procedures for ISO 10303 application protocols are given in [13].

NOTE 5 – The development practices for ISO 10303 application protocols are given in [13].

NOTE 6 – The ISO 10303 application protocol development methodology is designed to develop application protocols that are independent of given organisation(s) or enterprise(s). For this reason, none of the constituent models of an application protocol contain organisation or enterprise-specific factors.

14.1 Application protocol scope

A scoping statement is developed through analysis of the stated industrial requirements that are to be satisfied. This statement identifies the key elements of the application protocol's application context.

NOTE 1 – The key elements of an application context are given in clause 9.1.

The identification of sample product data, together with usage scenarios that are pertinent to the application context, are used as aids in the definition of the detailed information requirements of the application protocol and the development of conformance classes and abstract test suites.

NOTE 2 – The development procedures for the scopes of ISO 10303 application protocols are given in [13].

NOTE 3 – The development practices for the scopes of ISO 10303 application protocols are given in [13].

14.2 Application activity model

The scope of an application protocol is refined through the development of an application activity model. At each level of decomposition in the application activity model, every activity, and associated inputs, controls, outputs and mechanisms, are examined and identified as being either in or out of the scope of the application protocol.

Analysis of the mechanisms that are included in an application activity model is required to ensure that they are relevant to the viewpoint and purpose of the application activity model.

EXAMPLE 16 – The resources and tools that are used in a given activity are examples of mechanisms.

EXAMPLE 17 – Mechanisms are frequently organisation or enterprise-dependent. Such dependencies are not suitable for inclusion in an ISO 10303 application activity model. Only those mechanisms that are independent of organisation or enterprise are included in an ISO 10303 application activity model.

An application activity model identifies activities and key information flows that are in the scope of the application protocol. As the application activity model and information requirements become more detailed, the scope of the application protocol may be refined or modified as necessary.

NOTE 1 – The development procedures for ISO 10303 application activity models are given in [13].

NOTE 2 – The development practices for ISO 10303 application activity models are given in [13].

14.3 Application reference model

The application reference model of an application protocol is based upon its scope and application activity model.

The application reference model is modelled only to the level necessary to convey the information that is important from the application experts' point of view.

NOTE 1 – The graphical presentation of the application reference model aids understanding and review of the information therein.

The application reference model undergoes validation to ensure the following:

– that it is self-consistent;

– that it is consistent with the scope of the application protocol.

NOTE 2 – Information modellers validate the application reference model for self-consistency; this activity is sometimes referred to as "integrity testing".

Experts in the application context of the application protocol validate the application reference model with respect to the scope; this activity is sometimes referred to as "fitness testing".

The validation of the application reference model is a critical and resource-intensive activity. Complete model validation of a complex application reference model is impractical. It is usually evident from the development of the application activity model that recurring demands for the same information exist. These facts can be used to prioritise the parts of the application reference model to be validated. The objective of application reference model validation is to provide an acceptable level of confidence that it is complete and correct.

The example product data, and associated input and output information requirements, used in the initial scope and requirements definition of an application reference model are used for building the usage tests for its validation. Usage tests document typical operations for creating or accessing the given product data. The set of application reference model usage tests are defined to ensure coverage of the following:

– application context and functional requirements;

– the information defined in the application reference model;

– possible combinations of product representations.

NOTE 3 – Difficulty in defining a meaningful usage test which exercises an application object may indicate that the application object is not needed.

Application reference model validation, at a minimum, includes paper-based populations and reference path analyses of the application reference model.

NOTE 4 – A reference path is a route that an appropriate application system uses to access its input or output data when it is formatted according to the application reference model.

EXAMPLE 18 – One method of validating an application reference model is to build a prototype implementation which closely matches, if not replicates, the constructs of the application reference model. This prototype is then populated to test its ability to accommodate representative product data from the application context. Representative usage tests in the form of queries are posed upon

these populations to indicate whether or not the application reference model is sufficient to support the processes that are within the scope defined in the application activity model.

Detailed validation of the application reference model provides feedback on the information structures and requirements defined in the model. Iterations occur between the population and reference path analysis of the application reference model and its development. Further, as the application reference model is refined, traceability and consistency is maintained between the scope, application activity model, and application reference model.

NOTE 5 – The development procedures for ISO 10303 application reference models are given in [13].

NOTE 6 – The development practices for ISO 10303 application reference models are given in [13].

14.4 Unit of functionality

As the application reference model is developed, the constructs which correspond to each unit of functionality are grouped together so that they are identifiable. A list of the units of functionality with definitions is maintained. This list includes the activities of the application activity model that require each unit of functionality.

NOTE 1 – The documentation of units of functionality may facilitate the integration of application reference models.

NOTE 2 – The development procedures for ISO 10303 units of functionality are given in [13].

NOTE 3 – The development practices for ISO 10303 units of functionality are given in [13].

14.5 Mapping table

The mapping table is developed and documented during the development of the application interpreted model. A record is kept of the usages of the integrated resources that are made during the interpretation process. This record is the source material for the mapping table. It is used to specify the relationship between each information requirement in the application reference model and its corresponding construct in the application interpreted model.

If a path of entity references in the application interpreted model is to be followed to satisfy a particular requirement in the application reference model then the entire path is given in the mapping table. A reference path is required in the following situations:

– when an attribute in the application reference model and the entity to which it belongs are not contained in the same entity in the application interpreted model;

– when an attribute in the application reference model is developed to a greater level of detail than in the integrated resources;

NOTE 1 – In this case, the reference path is provided so that the complete semantics (including the relationship of the attribute to the entity in the application reference model) is represented in the mapping table.

– when a group of application interpreted model entities is required to satisfy a single relationship that is defined in the application reference model;

– when subtypes are created in the application interpreted model.

NOTE 2 – In this case, the reference path shows the supertype from the integrated resources and any mapping rules or choices that are specified.

NOTE 3 – An example mapping table is given in annex D.

The mapping table is reviewed to ensure that it is complete and correct.

NOTE 4 – This review requires expertise in both the ISO 10303 data specifications and the application context.

The mapping is complete when each application reference model construct has an equivalent construct(s) in the application interpreted model.

NOTE 5 – The development procedures for ISO 10303 mapping tables are given in [13].

NOTE 6 – The development practices for ISO 10303 mapping tables are given in [13].

14.6 Application interpreted model

The development of the application interpreted model is based upon the content of the application reference model. Interpretations of the integrated resources are made to satisfy the requirements of each application object in the application reference model.

14.6.1 Definitions relating to interpretation

14.6.7 relationship subsetting interpretation: the process by which elements of the integrated resources that are applicable in a given application context are defined.

EXAMPLE 19 – The specification of the subtypes of a given entity that are mandatory in the given application context is an example of relationship subsetting.

14.6.8 role establishment interpretation: the process by which application-context-specific relationships between templates from the integrated resources and interpreted integrated resource constructs are defined.

EXAMPLE 20 – The definition of relationships between the management resources and other elements of an interpretation of the integrated resources is an example of role establishment.

14.6.9 relationship constraint interpretation: the process by which integrated resource relationship instances that are applicable in a given application context are defined.

EXAMPLE 21 – An integrated resource construct may include an aggregate of 1 to many elements but only aggregates with 2 or 3 elements may be applicable in the given application context. The restriction of the range of the aggregate is an example of relationship constraint.

14.6.10 attribute value constraint interpretation: the process by which integrated resource attribute instances that are applicable in a given application context are defined.

EXAMPLE 22 – Only specific values of given attributes of given integrated resource constructs may be applicable in a given application context. The restriction of the range of the attribute values is an example of attribute value constraint.

14.6.2 Interpretation methodology

Development of an application interpreted model includes the following activities (see clause 14.6.1 for definitions of terminology):

a) identification of the integrated resource constructs that correspond to given application object(s);

NOTE 1 – To represent the required functionality using integrated resource constructs, each of the application objects is examined to find a corresponding construct or group of constructs in the integrated resources. At this stage of the interpretation process, only those constructs which satisfy an information requirement directly are identified.

b) identification of integrated resource constructs that require specialisation;

NOTE 2 – There may be application reference model constructs whose semantics are covered by the semantic intent of a given integrated resource construct. Such integrated resource constructs require the addition of constraints if an exact semantic correspondence is to be provided. At this stage of the interpretation process, only those constructs requiring specialisation in the application interpreted model are identified.

c) specialisation of integrated resource constructs with partial correspondence;

(1) semantic specialisation: a new subtype is specified and described using natural language as a refinement of the supertype but further constraints are not added to the supertype;

NOTE 3 – The new subtype entity may contain derived attributes that correspond to the attributes of the supertype but with attribute names that are specialised to be consistent with the semantic requirements of the application reference model.

(2) syntactic specialisation: a new subtype is specified, described using natural language as a refinement of the supertype and further constraints are added to the supertype.

NOTE 4 – The development procedures for ISO 10303 application interpreted models are given in [12,13].

NOTE 5 – The development practices for ISO 10303 application interpreted models are given in [12].

NOTE 6 – The *EXPRESS* usage guidelines for ISO 10303 application interpreted models are given in [12].

Once an initial correspondence between application objects and assertions and the integrated resource constructs has been established, an *EXPRESS-G* representation of the application interpreted model is produced.

NOTE 7 – The *EXPRESS-G* diagrams are useful in developing the application interpreted model *EXPRESS* listings, both the short and annotated.

The application interpreted model *EXPRESS-G* diagrams include all *EXPRESS* entities, enumeration types and select types.

Comprehensive validation of a complex application interpreted model is a resource intensive activity. The objective of application interpreted model validation is to provide an acceptable level of confidence in its completeness and correctness.

Potential overlaps between application protocols manifest themselves in the form of common information requirements, similar units of functionality, or correspondence between application reference models. Knowledge of the area and status of other application protocols provides this information. The application interpreted model development method requires that, when overlaps occur, either application interpreted constructs are used or common interpretations are made.

14.7 Conformance class

An Application Interpreted Model, as discussed above, provides the normative specification for data to be exchanged between computer applications. This provides the scope and boundaries for implementations of product data exchange that conform to ISO 10303, and also the scope and boundaries for testing implementations. In order to avoid problems caused by implementations of vendor-specific subsets, ISO 10303 Application Protocols require completeness of implementation, ie. that an interface conforming to the Application Protocol should support every entity, attribute and constraint specified in the AIM.

In order to meet the needs of differing computer systems used within a given industrial application, whilst maintaining consistency of implementation and testing, two or more Conformance Classes may be defined for an AIM. A conformance class defines a subset of the AIM that may be used as the basis for implementation and testing. These subsets define the minimum conforming implementation based on the AIM; implementations based on any other subsets are not considered to be conforming. The completeness requirement then applies within each conformance class.

Conformance classes are developed through analysis of the usage scenarios identified in the initial phase of the development of the Application Protocol, and through understanding of the capabilities of the computer applications that are expected to support the Application Protocol. Each conformance class defines a fixed boundary for the scope of an implementation; this is determined on the basis of defining a subset of total capability of the Application Protocol that is practical to implement whilst not comprising the industry needs that define the purpose for the existence of the Application Protocol.

NOTE 1 – The development procedures for ISO 10303 conformance classes are given in [13].

NOTE 2 – The development practices for ISO 10303 conformance classes are given in [13].

15 Integrated resources

Integrated resource constructs are developed to support the development of application interpreted models. Requirements are established in the form of draft resource models which are and then integrated with the existing integrated resources. In this way the integrated resources are extended to meet new application requirements as they arise. The purpose of the integration process is to maintain the integrated resources as a single, self-consistent and integrated model.

Analysis of the information requirements specified in an application reference model may identify the need for extension to the integrated resources.

NOTE 1 – This approach to the extension of the integrated resources represents the mature phase of ISO 10303 development. Previously, complete, existing draft resource models have been integrated with the core generic product description resource.

As with application interpreted model development, resource integration is a process which involves the interaction between the experts in the discipline covered by the draft resource model and experts in the ISO 10303 integration process and integrated resources.

15.1 Definitions relating to integration

15.1.11 analysis with respect to conceptual uniqueness: evaluation of two or more data specifications to detect redundancies and conflicts between the concepts that are represented therein. The following types of conflict are identified:

– naming conflicts involving either homonyms, where the same name is used for different concepts, or synonyms, where different names are used for the same concept;

– structural conflicts involving the following:

- type conflicts where the same concept is represented using different data types;
- dependency conflicts where the same relationship between entities has different cardinalities;
- definitional conflicts where different mandatory attributes exist for the same concept;
- behavioural conflicts where different rules regarding data integrity are specified.

15.1.12 analysis with respect to functional adequacy: evaluation of a data specification to ensure that each of its elements is traceable to an element of established requirements of an information area in the ISO 10303 data specification architecture.

15.1.13 achievement of modelling consistency: the removal of inconsistencies in modelling style across data specifications that are undergoing integration.

NOTE 1 – Of particular importance for modelling consistency in ISO 10303 is the identification of existence dependency among related concepts. Existence dependency involves a relationship between concepts where an instance of one concept is incomplete without the presence of an instance of another. The modelling approach taken in ISO 10303 is to have the dependent concept reference the

concept upon which it is dependent. This modelling principle is consistent with modularisation such that the phased development of the standard involves extensions that build upon existing concepts.

NOTE 2 – Consistency within the integrated resources is ensured through the use of template structures.

EXAMPLE 23 – The template structure that is used to capture the semantics of relationships between similar entities is given in ISO 10303-41.

NOTE 3 – Consistency in the way in which the management resources are specified in the integrated resources and interpreted in application interpreted models (through role establishment) is also ensured through the use of templates. These templates are specified in the integrated resources using the *EXPRESS* ABSTRACT SUPERTYPE construct. They are interpreted in application interpreted models using the *EXPRESS* SUBTYPE construct.

15.1.14 resolution of identified conflicts: the removal of conflicts between data specifications that are undergoing integration.

NOTE 4 – Techniques used to resolve conflicts include the creation of a generalised construct to accommodate conflicting requirements, creation of a specialisation of a construct, and removal of constructs for later resolution when sufficient knowledge is available for confident resolution.

15.1.15 achievement of non-redundancy of constructs: the removal of redundancies across data specifications that are undergoing integration.

15.1.16 conceptual nature of construct: integrated resource constructs are generic and conceptual in nature to provide shareability among multiple application contexts and implementability within a diverse heterogeneous environment of computer platforms. The integration process ensures that constructs are conceptual in nature. Constructs in the integrated resources convey semantics that logically describe product data concepts. These constructs do not include ideas or mechanisms that are motivated by convenience in practices, computer technologies, or efficiency requirements for implementation.

15.1.17 placement in the data specification architecture: the logical placement of data specification constructs in the data specification architecture. Changes to either the constructs or the architecture may be made during the integration process when incompatibilities are identified.

NOTE 5 – Over time, because of the impact on the already integrated constructs, it is advisable to consider changes in a proposed additional construct rather than in the data specification architecture, although adding constructs rather than changing the architecture may result in requirements not being met or might affect the way in which requirements are met.

15.1.18 entity, relationship, and attribute specification : the specification of entities, relationships and attributes is the primary integration issue. Entities and their attributes are specified consistent with the scope of the integrated resources. Relationships are specified such that references are based upon existence dependency. Common canonical structures are employed

for consistency and efficiency. Entities, relationships and attributes are added to resolve voids for completeness of constructs.

15.1.19 generalisation: the modification of concepts to be appropriate for the scope and context of the integrated resources.

NOTE 6 – The scope and context of the integrated resources requires that the represented concepts are free of any specific product data application context.

Concepts that elaborate upon the semantics of more generic concepts that have been previously integrated are specified as specialisations.

15.1.20 subject-area constraint: a constraint that includes relationship constraints and attribute-value constraints. Relationship constraints establish subject-area rules for references between concepts. Attribute-value constraints establish subject-area rules for data consistency and integrity.

15.1.21 modularisation: in the context of the ISO 10303 integration process, the division of a draft resource model into a number of manageable and conceptually consistent constructs.

15.1.22 integrated resource constraint : a constraint that establishes rules specific to the scope and context of the integrated resources. Such constraints typically cross module boundaries.

EXAMPLE 24 – Examples of such constraints are the rules that ensure consistency between the definition and representation of product data in [ISO 10303-41,ISO 10303-43].

NOTE 7 – Integrated resource constraints cross the boundaries between the modules of the integrated resources whereas subject-area constraints do not.

15.2 Integration methodology

The integration methodology includes the following activities (see clause 15.1 for definitions of terminology):

a) semantic analysis – draft resource models are analysed to determine underlying meaning;

NOTE 1 – The concepts represented by the draft resource model are compared with the concepts represented by the integrated resources. Constructs are evaluated in terms of conceptual uniqueness and functional adequacy. Every integrated resource construct is established to fulfil a particular application context requirement.

Associations among resource constructs are identified for possible interfacing in a later stage of the integration process.

NOTE 2 – The semantic analysis of draft resource models includes analysis with respect to conceptual uniqueness and analysis with respect to functional adequacy.

b) harmonisation – provides non-redundancy of constructs, resolution of identified conflicts, and modelling consistency;

NOTE 3 – Constructs are conceptually aligned by resolution of detected conflicts.

NOTE 4 – The result of the harmonisation process is a minimal set of consistent constructs that are ready for structuring.

NOTE 5 – Harmonization includes the achievement of modelling consistency, the resolution of identified conflicts and the achievement of non-redundancy of constructs.

c) placement in the data specification architecture;

NOTE 6 – The ISO 10303 data specification architecture is described in annex B.

d) structuring – constructs are structurally aligned;

NOTE 7 – In the ISO 10303 integration process this is achieved modelling a new concept that supports the semantics required by both the draft resource model and the existing integrated resources.

NOTE 8 – It is within this activity that voids are detected and resolved. Structuring provides completeness, structural consistency and structural precision with respect to semantic intent. Structuring involves the following:

- entity, relationship and attribute specification;
- generalisation;
- the specification of subject-area constraints;
- modularisation;
- the specification of integrated resource constraints.

e) interfacing – constructs are conceptually and structurally related to other constructs within the integrated resources.

NOTE 9 – Interfacing maintains modularisation of constructs with a minimum of inter-construct references.

References between constructs are controlled to ensure consistency and manageability of the specialisation. References between concepts that are in different modules are controlled by the data specification architecture.

NOTE 10 – *EXPRESS* schemas are the modules of the integrated resources.

NOTE 11 – Consistent with the architecture, the more specialized constructs reference the more general constructs. Existence dependency rules determine reference directions between constructs. The controlled references minimize the impact of change and thus minimize upward compatibility issues.

NOTE 12 – The integrated resources may be extended to support the information requirements of a given application reference model without the development of an intermediate draft resource model. This approach requires the same set of expertise as the one given here but the time expended is reduced since superfluous draft resource models are not established. However, practical considerations may detract from this approach. For example, the establishment of a draft resource model allows consensus between area experts to be reached before integration is attempted.

NOTE 13 – A draft resource model or application reference model is a set of information requirements that the extended integrated resources are to satisfy rather than a model that is to be integrated with the existing integrated resources. Thus, the integration process becomes an extension process.

NOTE 14 – The development procedures for ISO 10303 integrated resources are given in [11].

NOTE 15 – The development practices for ISO 10303 integrated resources are given in [11].

NOTE 16 – The *EXPRESS* usage guidelines for ISO 10303 integrated resources are given in [15].

16 Application interpreted constructs

This clause describes methods for the development of the following:

- the scope and content of an application interpreted construct;
- the usage of an application interpreted construct by an application interpreted model.

16.1 Application interpreted construct development

The development of an application interpreted construct is based upon equivalent information requirements that are common to more than one application protocol. This equivalence of information requirements is detected either before or during the application interpreted model development process.

NOTE 1 – The detection of equivalent information requirements requires knowledge of other application protocols.

Two interpretations of a given group of integrated resource constructs may be syntactically equivalent. Such interpretations may also be semantically equivalent.

NOTE 2 – A pair of syntactically equivalent interpretations of the integrated resources may not be semantically equivalent.

EXAMPLE 25 – An application interpreted construct may define an interpretation of integrated resource constructs used to represent shape information for several different application protocols. These application protocols may also make use of interpretations of integrated resource constructs to satisfy requirements for product identification and versioning. These interpretations of integrated

resource constructs do not form an application interpreted construct unless the application protocol requirements for product identification and versioning are equivalent.

A key element of the strategy for the development, use, documentation and management of application interpreted constructs is the provision of common interpreted constructs that meet the requirements common to multiple application protocols. The semantic boundary of an application interpreted construct is determined by the requirements that are satisfied.

NOTE 3 – This provides stability and control over the total number of application interpreted constructs that require development.

The rules which control the content of an application interpreted construct are as follows:

a) at least one root entity is defined;

NOTE 4 – Each root entity is a SUBTYPE of an integrated resource entity.

b) no global rules are included;

c) all constraints which apply to more than one construct in an application interpreted construct are localised as *EXPRESS* domain rules in one of the root entities.

EXAMPLE 26 – The following *EXPRESS* illustrates the constructs found in an application interpreted construct.

```
*)
SCHEMA aic_xxx;
USE FROM ir_schema (ir_a);
ENTITY aic_a;
SUBTYPE OF (ir_a);
WHERE
   WR1 : -- aic specific constraints on a
   WR2 : -- etc.
END_ENTITY;
END_SCHEMA;
(*
```

NOTE 5 – The development procedures for ISO 10303 application interpreted constructs are given in [16].

NOTE 6 – The development practices for ISO 10303 application interpreted constructs are given in [16].

NOTE 7 – The *EXPRESS* usage guidelines for ISO 10303 application interpreted constructs are given in [16].

16.2 Application interpreted construct usage

All application protocols which share a common information requirement use the appropriate application interpreted construct in its entirety.

NOTE 1 – The phased development of ISO 10303 application protocols may result in a delay in the manifestation of an application interpreted construct in all of the application protocols that make

use of it.

EXAMPLE 27 – Application protocol'X' has been developed and standardized as a part of ISO 10303. During the subsequent development of application protocol'Y', an equivalence between requirements within application protocols 'X' and 'Y' is identified and the corresponding application interpreted construct developed. This application interpreted construct is referenced explicitly by application protocol'Y'; however, it will only be referenced by application protocol'X' when application protocol'X' is next revised.

NOTE 2 – An application interpreted construct is included in an application interpreted model through the *EXPRESS* 'USE FROM' keywords.

NOTE 3 – The *EXPRESS* 'USE FROM' keywords are used for purposes other than the inclusion of application interpreted constructs in application interpreted models. The use of the *EXPRESS* 'USE FROM' keywords for the inclusion of an application interpreted construct can be identified by the name of the included schema which contains the 'AIC' acronym.

An application interpreted construct is always used in its entirety. Selection from the constructs within an application interpreted construct is is not permitted. Rules to constrain populations of application interpreted constructs within a given application interpreted model are permitted in the following cases:

– *EXPRESS* local and global rules may be used to govern the valid population of constructs in the application interpreted construct;

– constraints to limit the relationships amongst application interpreted constructs when more than one application interpreted construct is used in a single application interpreted model.

NOTE 4 – The procedures for the use of application interpreted constructs in ISO 10303 application protocols are given in [16].

NOTE 5 – The practices for the use of application interpreted constructs in ISO 10303 application protocols are given in [16].

NOTE 6 – The *EXPRESS* usage guidelines for the use of application interpreted constructs in ISO 10303 application protocols are given in [16].

17 Abstract test suites

The method for the development of abstract test suites includes the following activities:

a) generation of the appropriate application reference model test purposes;

b) generation of the appropriate application interpreted model test purposes;

c) generation of other test purposes;

EXAMPLE 28 – ISO 10303-201 references ISO 3098-1 for use as its predefined text font.

d) definition of abstract test cases to reflect test purposes;

e) documentation of input specifications and reference results;

f) documentation of verdict criteria for each test case.

NOTE 1 – The development procedures for ISO 10303 abstract test suites are given in [13].

NOTE 2 – The development practices for ISO 10303 abstract test suites are given in [13].

Section III: Approaches to the implementation of ISO 10303

18 Implementation principles

This clauses specifies the principles of the implementation of ISO 10303 for product data exchange that the ISO 10303 architecture and development methodology has been established to support.

18.1 Fundamental assumptions

The following are fundamental assumptions:

a) the original ANSI/SPARC three-layer architecture included the implementation layer as the 'lowest' layer – this approach enables the form of each implementation method to be defined without knowledge of its contents when data is exchanged or instanced;

b) the product data aspect of any implementation of ISO 10303 for data exchange is required to be efficient and computer-independent and should facilitate conformance (and other types of) testing.

18.2 Fundamental concepts

The following are fundamental concepts:

a) An implementation of ISO 10303 for product data exchange is defined as the combination of a particular application protocol with a single implementation method.

b) All implementation methods necessarily contain:

- a formally-defined syntax;
- a mapping from Express constructs onto the implementation method.

c) some *EXPRESS* constructs are not relevant for a particular implementation method, and so no mapping is specified in the part for these constructs.

EXAMPLE 29 – Comments in the host *EXPRESS* schema for a particular entity declaration do not need to be translated into a physical file whenever an instance of that entity is required; consequently, there is no mapping for the *EXPRESS* comment construct in the physical file implementation method.

d) given the general mapping from each type of *EXPRESS* construct it is possible to deduce how each entity defined in an application interpreted model will appear on the physical medium.

18.3 Implementation approaches

The remainder of this clause addresses the questions that have been raised on how ISO 10303 is to be implemented. Also discussed are some of the popular issues concerning data exchange and data sharing, and the use of multiple application protocols in a single implementation. Many believe that data sharing is defined as the use of multiple application protocols in a single implementation. As such, some of the popular proposed solutions are focused on how to integrate application protocols into a single schema to solve the data sharing issue. In fact, to effectively address these related but independent issues, they need to be defined and discussed separately.

The issues associated with implementation are separated into the following areas:

a) data exchange and data sharing implementation using ISO 10303 application protocols;

b) implementations using a single ISO 10303 application protocol compared with those using multiple ISO 10303 application protocols;

c) depending on the implementation objectives, there may be many valid and useful ISO 10303 implementation approaches and system architecture;

The remainder of this clause is organized into sections to address these points.

18.3.1 Data exchange and data sharing

The ISO 10303 architecture and development methodology are designed to support the objective of the standard: that the standardized descriptions of product data are suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases.

In order to assess the nature of support for these in ISO 10303, it is necessary to first establish exactly what they mean and how they are different.

NOTE 1 – Examples illustrating differences between product data exchange and product data sharing are given in annex D.

18.3.1.1 Data exchange and ISO 10303

In a modern computing environment, data exchange is commonly understood as the exchange of neutral format data files between computer systems. A sending system translates data from its internal format and encodes it into an established neutral format. This file is then transferred to the receiving system where the data is translated into the internal format of the receiving system. The architectural components which comprise a data exchange implementation include:

- sending system translator (pre-processor) to generate neutral data file
- transport mechanism for sending neutral file to receiving system
- system translator (post-processor) to convert data file to internal format

There are many well understood transport mechanisms available for distributing a data file, including electronic mail and File Transfer Protocol (FTP).

In the banking analogy, the banking statement, including both the data and the data context, represented the neutral data file. In traditional computer-based data exchange, the neutral data file usually contains only the data, not the context. The context of the data, or how the data values relate to meaningful concepts, is often defined implicitly by the data's location in the format file. Both the sending and receiving systems depend on a priori knowledge of the file format and its relationship to the data context. If this were true in the banking analogy, the account statement would contain only numbers; the bank and the customer would need to agree beforehand which numbers correspond to which account categories (i.e. line 1 would contain the balance, line 2 would contain disbursals during the reporting period, etc.)

ISO 10303 supports data exchange through the ISO 10303 physical file format. Each ISO 10303 application protocol provides an application interpreted model which serves as an explicit standardized data specification for an established application context. It is the application interpreted model which provides a documented explanation of the context (scope) and meaning (relationships) of the data to be exchanged. The application interpreted model is used, along with an encoding algorithm, to produce ISO 10303 physical files which contains both the data and its associated context for effective and flexible communication between computing systems.

18.3.2 Data sharing and ISO 10303

To be successful in accomplishing data sharing, many aspects of information technology are required to serve as various elements of a data sharing process. Among those, controlled data access is a key element. Data access is the interface mechanism whereby a computer program can read or update the selected data in a data store. In the data sharing scenario, some portion of the database is intended to support joint usage and ownership. To successfully share data, the data access mechanism must allow multiple application systems to read, store, and manipulate the shared data. Traditionally, data sharing systems are implemented as client-server systems, or more recently as server-server (federated) systems. Multiple cooperating applications may have multiple data stores. As in the banking analogy above, the communication protocols to achieve data sharing are generally more complicated than those of data exchange. The information context is embedded in any one application's knowledge of external systems, their physical schemas, and the steps required to access/update the external store. Effective data sharing requires a mapping of these different stores to a single data specification.

ISO 10303 supports a data sharing environment by contributing the application interpreted model and associated conformance classes, data instantiation rules and constraints that are specified in the mapping table, to serve as the single data specification for the application systems that need to share data.

Because ISO 10303 is developed as a communication standard, it is not intended to serve as an efficient and complete data structure for storage implementation. However, an application protocol's application interpreted model can serve to provide the necessary navigation paths for data access. One of the mappings is with a user's conceptual view of the information (context), represented abstractly in an application protocol's application reference model. The other mapping is to the physical databases where the shared data are stored.

The design of a data sharing implementation is required to consider the system architecture and computing environments (e.g., hardware, software, and development tools) of all application sys-

tems that need to share data. The system design must address the unique interactions between the controlling structure and all its component systems. The design strategy varies depending on the component systems, the computing environment, and performance considerations.

18.3.3 Open data and ISO 10303

Much has been written concerning the goal of Open Systems. In the 1980's the trend toward open systems produced some successes in the area of operating systems with the introduction of POSIX standards. These standards identified a common set of interfaces to which compliant operating systems must subscribe.

It has been theorized that an open data environment would provide significant advantages for users of information-intensive systems. The vision of an open data environment is to standardize on a single conceptual representation of information and associated interface mechanisms within an application area. Application developers would be required to write application software to conform to the representation and interfaces defined in the standard. The goal of this environment would be to allow a customer to plug and play competing applications from different vendors without expending significant resources to migrate the underlying application data form one format to another.

While the feasibility of such an approach requires significantly more study, it is clear that ambitious goals of an open data environment require a standardized data specification for information within an application area. It is hoped that the methods and approach of ISO 10303, built upon the concepts of standardized data specifications, could be leveraged to supply the necessary data requirements for an open data environment.

18.4 Single or multiple application protocol implementation

The data exchange or data sharing implementation may use a single application protocol or have the need to use multiple application protocols. Data exchange and data sharing implementation system architectures and architectural components remain the same if the communication controlling definition is always a single schema. This single schema may be the application interpreted model of a single application protocol or a merged schema from several application interpreted models of different application protocols. Figure xx depicts the relationship of the communication controlling schema and the system architectures.

For the same reason, the strategy of using a single application protocol or using multiple application protocols is not dependent upon the transaction forms, e.g., data exchange or data sharing or the characteristic of the implementation, e.g., on-line access or the exchange of a physical file. But rather it is dependent upon the scope and application context of the data required in the implementation. So, issues associated with using multiple application protocols are entirely different issues associated with data exchange or data sharing implementation architectures. This does not mean that one can not design a system architecture that can accommodate both functions.

This section addresses the issues and strategies associated with implementations using a single application protocol or using multiple application protocols.

18.4.1 Single application protocol implementation

Using a single ISO 10303 application protocol for an event of data exchange or data sharing, ISO 10303 clearly specifies the conformance criteria. The conformance criteria for the implementation of a single application protocol includes the conformance class, the data specification, interpretation and mapping between application reference model and application interpreted model, and the abstract test suite. A conforming implementation using a single application protocol must be able to provide data instances that can satisfy:

– subset of application interpreted model EXPRESS schema structure and definitions (Clause 5.3 of the application protocol document) that belongs to an application protocol specified conformance class (Clause 6 of the application protocol document);

– the instantiation constraints and rules specified in the application reference model to application interpreted modelmapping table (Clause 5.2 of the application protocol document); and

– the abstract test suites (the application protocol's corresponding 300's Part).

18.4.2 Multiple application protocol implementation

The data exchange or data sharing implementation of multiple application protocols does not require different implementation system architecture than that of a single application protocol. What it does require are:

– established guidelines on if and how application interpreted model schemas from different application protocols can be merged; and

– define the conformance criteria.

18.4.2.1 application interpreted model schema merge

Each application interpreted model is the usage (i.e., interpretation) of integrated resource definitions in a specialized context. For example, integrated resource defines an entity called shape_aspect to be an aspect of a product's entire shape, which is a kind of property that a product has. Under the context of an application protocol which is intended to communicate the description of shape features of mechanical parts for machining, the integrated resource generic definition of shape_aspect is interpreted to be a machining feature that has specific parametric descriptions. For example, a chamfer is defined by a transition between two edges of a surface with a flat cross section. Associated with the chamfer shape_aspect, there is an angle and an offset length between two edges of a surface where the angle applies to specified.

The same integrated resource entity shape_aspect, under the context of a different application protocol, e.g., for electrotechnical system design, is used to define the termination of a device that can be connected to the termination of another device. The parametric shape description is not relevant under this context, but rather its spatial location is important design information.

An implementation of the application protocol to communicate shape aspects to support mechanical fabrication applications may only chose to use the single application protocol that

defines shape aspects under the context of machining feature. An implementation that is interested to consolidate the usage of shape aspect from both the perspectives of machining features and connectivity of device terminations would require the merging of these two independently designed application protocols as a composite or consolidated schema for a repository structure. Both sets of rules/constraints that govern the instantiation of shape_aspect entity in the consolidated schema are valid, however they are valid on different sets of the population.

In order to merge two or more application interpreted model schemas into one consolidated schema for expanding the scope and context of any single application protocol, some of the technical issues are identified as follows:

– Alignment of application context

The fact that the same entity appears in multiple application protocols does not mean that they can or is meaningful to be automatically merged into a single model. The relationships must be established between the usages of that entity by the two application protocols.

– Explicit interface definitions

When an application protocol is being development, explicitly identify its potential expansion of scope for the purpose of leveraging other existing application protocols or future application protocols. For example, an application protocol that is focused on a specific area, such as electrotechnical plant design does not include mechanical perspective of shape geometry for the devices. The application protocol may define another product_definition_context that is an additional but related application context, e.g., mechanical design. Through this common product_definition_context, it allows the merge of the application interpreted model schema to another application interpreted model schema which contains mechanical design context and shape geometry information.

This additional and explicit addition of an intended interface definition allows two or more application protocols to be merged at the product definition entity.

– Use of application interpreted model as a definitional module

Two or more application protocols may have overlap contexts and information requirements. The overlap is defined as an AIC. This same AIC is used as a common definitional module for both application protocols. The application protocols that have these overlaps can include the same AIC schema. For example, Part 202 has a draughting application context and Part 204 has the mechanical design shape context share the same usage of the integrated resource definitions of presentation of visual styles of geometric objects.

Each AIC schema is defined with a root entity which specifies all integrated resource usage rules/constraints. If an implementation intends to merge two or more application interpreted modelschemas, a method may be to create a new schema which includes two or more application interpreted modelschemas the same way an application interpreted model schema includes AIC schemas. Using this method, a root entity should be defined for each of the application interpreted model schema to convert the single application interpreted model's global rules into local rules of the application interpreted model's new root entity. This

method manages the use of specific rules that each application interpreted model may have into a global context of the merged multiple application interpreted model schema.

– Syntactic Issues

In addition to the semantic issues of merging application interpreted model schemas, there are a number of syntactic issues.

– EXPRESS language

In a merged schema, each application interpreted model schema name in the EXPRESS fully qualified name must be changed into the merged schema name.

– Part 21

Within a Part 21 file, each instance has a unique instance identifier. Part 21 uses this instance identifier for reference. The design principle assumes that each instance is unique. This design does not offer the proper mechanism for references of common instance across files. If two files contain the same instance identifiers, one must be changed to avoid misreferencing. If two application interpreted modelschemas are merged into a new schema, physical files that are based on each of the single application interpreted model schemas, can not be merged effectively. Because common instances are not identifiable, the data referential integrity can not be maintained.

18.4.2.2 Conformance criteria

There are two alternatives for ISO 10303 architecture to address implementations using multiple application interpreted model schemas. One is to provide guidelines only on how it can be done. This alternative leaves open the choices of application interpreted model schemas to be merged for individual implementation to decide. No specific conformance classes are identified for the merged schema. A conforming data communication, that is for both data exchange and data sharing, is based on the identified conformance classes within each individual application protocol.

The other is to standardize the conformance criteria for implementations of merged multiple application interpreted model schema. The conformance classes defined for each individual application protocol may require significant modification to be useful for the merged multiple application interpreted model schema. If ISO 10303 were to support a conforming implementation of multiple application protocols, methods for the development of conformance classes and new abstract test suites must be established.

All of the currently available ISO 10303 parts: EXPRESS language, Part 21 file format, conformance criteria, and conformance testing specifications are designed to support the single application protocol implementation. If ISO 10303 were to provide conformance criteria for implementations of multiple application protocols, issues associated with EXPRESS language, Part 21, SDAI, conformance classes, and conformance testing must also be addressed.

Figure 5 – Generic system function

18.5 System architecture

Based on the ISO 10303 architecture and method, there is a generic system function that is required for all ISO 10303 application protocol implementations. This generic system function is a data management function. Figure 5 depicts this generic data management function. There are four key components:

a) A conceptual schema processor that provides the capability to store, parse, and understand the conceptual schema.

b) A schema mapper which maps the conceptual schema definitions and structure to a different schema that is locally used for the application logic. This second schema may be the physical database schema or an application user interface data structure.

c) a local or physical schema processor that provides the capability to store, parse, and understand the schema that describes the locally managed data.

d) a data access function which reads and writes to the physical data store, physical data file, or user interface data structure based on the local schema and the local application system logic.

When a request to access data is received, the requester issues the request using the conceptual schema definitions. This conceptual schema definition is received by the conceptual schema processor to verify that the definition is valid. The validated request is passed to the schema mapper to map to the local schema. The schema mapper then passes the request using the locally managed schema definition to the local schema processor. The local schema processor validates the request against the local schema definition. Once validated, the local schema processor issues the request to the data access mechanism. The data access mechanism is responsible for the retrieval, update, and storage of the physical data.

Depending on the implementation objectives, this generic system function can be invoked to serve specific purposed. Figure 6 describes a data exchange system architecture between two application systems A and B. The exchanged data is in Part 21 file format. The conceptual schema is an application interpreted model_X.

Figure 6 – Data exchange architecture

Figure 7 depicts a data sharing system architecture. It is a combination of client/server system architecture and a federated system architecture that accommodate multiple servers and distribute the shared data among the servers. This system architecture also includes two application systems, C and D that are sharing data from a Part 21 file. This illustrates a significant consideration that Part 21 file, although are recognized traditionally as a data exchange implementation form, can also serve as the data store for shared data. In reverse, a database implementation does not necessary mean it is a data sharing implementation.

Figure 7 – Data sharing architecture

Annex A
(normative)

Information object registration

In order to provide for unambiguous identification of an information object in an open system, the object identifier

iso standard 10303 part(13) version(1)

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO 8824-1, and is described in ISO 10303-1.

Annex B
(normative)

The ISO 10303 data specification architecture

The integration framework of ISO 10303 is contained within the integrated resources. The integrated resources constitute a generic model of product data and are more abstract than the product data on which applications operate.

NOTE 1 – The integrated resources are generic in that they are intended to be applicable, through interpretation, to any product type. The integrated resources fall into two categories: the integrated generic resources and the integrated application resources. The integrated generic resources are also generic in that they are intended to be used, through interpretation, in any application context. On the other hand, the integrated application resources are intended to be used, again through interpretation, in application contexts that support selected application functionalities.

EXAMPLE 30 – The **product** entity from ISO 10303-41 is an integrated generic resource that is used in all application contexts. On the other hand, the **drawing_revision** entity from ISO 10303-101 is an integrated application resource that is used only in application contexts that include draughting.

The integration framework provides for the identification of a product with one or more specific disciplines as a frame of reference for the product's existence. Multiple groupings of definitions of the identified product may exist.

EXAMPLE 31 – The definitions of a given product may be grouped according to the version of the product to which they apply.

Multiple life-cycle-dependent views may exist for each grouping. Each view may be characterised by multiple properties. Any given property characterisation associated with a life-cycle-dependent view is only a portion of what is known about a product. These characterisations may be represented in multiple ways.

EXAMPLE 32 – The characterisation of the shape property of a product may be represented using either a boundary representation or a constructive solid geometry representation.

An application protocol contains an application interpreted model that uses the abstract constructs of the integrated resources. This is the portion of the application protocol that embodies the generic product data perspective of the application protocol. The application interpreted model is also that portion of the application protocol that is used as an element of the data exchange aspect of the data architecture. Conformance classes are specified that identify those portions of the application interpreted model that are accommodated by a computer system implementation if that system is to be judged to be conforming.

An application protocol also contains elements of the data usage aspect of the data architecture. These include the application activity model, scope, and application reference model. The level of abstraction used in developing these elements is quite different from that of the integrated resources and application interpreted model. Here the focus is on an application domain ontology that describes the application activities, information flows, application objects, and assertions relating these objects in the terminology and at the level of abstraction that is used in performing enterprise tasks. This is the portion of the application protocol that embodies its application domain perspective.

An application protocol also contains elements of the data specification aspect of the data architecture. This includes the application reference model, the mapping table and the application interpreted model. The application reference model is part of the data usage aspect and has an application domain perspective. The application interpreted model is part of the data exchange aspect and has a generic product data perspective. The data specification element that documents the transformation from one perspective (or level of abstraction) to another is the mapping table. The reason why ISO 10303 provides detailed descriptions of two different perspectives of the product data used by an enterprise application is that they are different models developed for different purposes:

a) the application reference model describes the product data of an application protocol as part of an application domain ontology where its definition uses concepts that are as they are used in performing enterprise tasks (described in the application activity model and scope);

NOTE 2 – This is intended to be useful for enterprise participants who come to ISO 10303 to understand the information that is covered by a given application protocol. It is in terminology and from a perspective with which they are familiar.

This is the data usage aspect of the application protocol.

b) the application interpreted model describes the product data of an application protocol in more abstract terms that are used by all application protocols for data exchange and relates to the data needs of application software.

The application interpreted model is not only more abstract than the application reference model but also more detailed in its representation of product data. Each application object and assertion of the application reference model is described in terms of the generic product data constructs that are used to provide this greater level of semantic detail. This provides the basis for understanding the relationship between product data exchanged using one application protocol and product data exchanged using another application protocol. If the product data of the two application protocols does not concern one or more of the same products then the data of the two application protocols is unrelated. If it does concern the same product then properties of the various views identified within the application protocols may be related to each other, through the data architecture, as being of the same product. This capability is an inherent aspect of the integration framework built into the integrated resources and manifested in the application interpreted model of an application protocol.

Annex C
(informative)

References

[1] ICAM Architecture Part II, Volume IV – Function Modeling Manual (IDEF0). 1981. Report number AFWAL-TR-81-4023, volume IV. June 1981. Available from: Mantech Technology Transfer Center, WL/MTX, Wright-Patterson Air Force Base, OH 45433-6533, USA.

[2] Integrated Information Support System (IISS). Volume V – Common data model subsystem. Part 4 – Information Modeling Manual – IDEF1 Extended. (IDEF1X). 1985. Report number AFWAL-TR-86-4006, volume V, part 4. November 1985. Available from: Mantech Technology Transfer Center, WL/MTX, Wright-Patterson Air Force Base, OH 45433-6533, USA.

[3] Verheijen, G.M.A. and van Bekkum, J. 1982. NIAM: an information analysis method. Journal: See: Olle, T.W. et al (1982), pp537-589.

[4] Tsichritzis, D. and Klug, A. (editors). 1978. The ANSI/X3/SPARC DBMS framework: report of the study group on database management systems. Information Systems, volume 3, pp173-191. Published by Pergamon Press Ltd, Oxford, UK.

[5] Olle, T.W, Sol, H.G. and Verrijn-Stuart, A.A. (editors). 1982. Information systems design methodologies: improving the practice. Published by North-Holland Publishing Company.

[6] Olle, T.W, Sol, H.G. and Verrijn-Stuart, A.A. (editors). 1982. Information systems design methodologies: a comparative review. Published by North-Holland Publishing Company.

[7] Halvorson and Palmer. ISO 10303 application protocols Status and Summary Report. ISO TC184/SC4 PMAG.

[8] Application protocol Integration Practices: Application interpreted construct Development. ISO TC184/SC4/WG4 Document N53. October 1992.

[9] ISO 10303 application protocol qualification manual. ISO TC184/SC4/WG4(P5) Document N502. November 1993.

[10] ISO 10303 Development Methods: Resource Integration and Application Interpretation. NIST IR. National Institute of Standards and Technology. DRAFT. March 1992.

[11] Guidelines for integrated resource Development. ISO TC184/SC4/WG10(P1) Document N37. September 1993.

[12] Guidelines for application interpreted model Development. ISO TC184/SC4/WG4(P3), document N302. September 1993.

[13] Guidelines for the development and approval of STEP application protocols. ISO TC184/SC4/WG4, document N66. version 1.1. January 1993.

[14] Supplementary Directives — Supplementary directives for the drafting and presentation of ISO 10303. version 2.3. March 1995.

[15] STEP EXPRESS Usage Guide. ISO TC184/SC4/WG5 N26. October 1991.

[16] Guidelines for application interpreted construct Development. ISO TC184/SC4/WG10 N??.

Figure D.1 – Existence dependence of ISO 10303 models

Annex D
(informative)

Examples

D.1 The framework of the integrated resources

The models that capture this framework embody the principle of existence dependence which ensures that all product information in a given exchange is related to an identified product and ultimately to an application context. This structure is summarised graphically in figure D.1.

NOTE 1 – The principle of existence dependence can lead to models that are at first sight counter-intuitive. For example, the application of existence dependence produces a generic product description resource where the definition of a shape includes a reference to the product that it is the shape of; a more intuitive approach is likely to result in the definition of a product including a reference to its shape. Simple analysis of this example shows that the existence dependent form of the model requires that a shape is related to a product.

ISO 10303 does not permit the existence of any product property data which is not related to a product.

Application element	application interpreted model element	Source	Rules	Reference path
Advanced_b_rep				
ADVANCED_B_REP	advanced_brep_representation	203		shape_representation => advanced_brep_representation
Authorisation				
APPROVAL	cc_design_approval	203	1,2	approval_assignment => cc_design_approval
date	date	41		cc_design_approval <= approval_assignment approval_assignment.assigned_approval -> approval <- approval_date_time.dated_approval approval_date_time.date_time -> date_time_select = date_and_time date_and_time.date_offset -> date
purpose	approval.purpose	41		cc_design_approval <= approval_assignment approval_assignment.assigned_approval -> approval approval.purpose

Table D.1 – Mapping table example

EXAMPLE 33 – Through the existence dependent structures in the ISO 10303 integrated resources, a collection of geometric data, such as collections of points, lines and curves, must be the representation of a property that is related to the definition of a product that has validity in some application context.

Thus the basic structure of the ISO 10303 integrated resources satisfies and enforces the principle that all product data be traceable to an industry need.

D.2 Example mapping table

Two units of functionality are given, **Advanced_b_rep** and **Authorisation**. The mappings of two application elements, **ADVANCED_B_REP** and **APPROVAL**, are provided. The mappings are described as shown in table D.1.

1. approval_requires_approval_date_time
2. approval_requires_approval_person_organisation

– The application element **ADVANCED_B_REP** maps to the application interpreted model entity **advanced_brep_representation**. The source column value denotes that the application interpreted model entity **advanced_brep_representation** is an application protocol specialisation, originating in ISO 10303-203. This specialisation requires a reference

path from the integrated resource entity from which it is specialised. The reference path denotes that the application interpreted model entity **advanced_brep_representation** is a subtype of the integrated resource entity **shape_representation**.

– The application element **APPROVAL** maps to the application interpreted model entity **cc_design_approval**. The source column denotes that the application interpreted model entity **cc_design_approval** originates in ISO 10303-203. This specialisation requires a reference path from the integrated resource entity to the specialised subtype. Rules 1 and 2 which are found at the end of the table constrain the use of the approval structure.

– The application element **APPROVAL** has an attribute **date** which maps to the **date** entity in the application interpreted model. The **date** entity originates in ISO 10303-41 as indicated in the source column. Since the attribute maps to an entity in the application interpreted model, a reference path is give from the entity **cc_design_approval** (this is the entity to which the application element **APPROVAL** was mapped) to the **date** entity (this is the entity to which the application reference model attribute **date** is mapped). The reference path is to be read as follows:

- **cc_design_approval** is a subtype of **approval_assignment**;
- **approval_assignment** has an attribute named **assigned_approval** that references the entity **approval**;
- **approval** is referenced by the attribute **dated_approval** in the entity **approval_date_time**;
- **approval_date_time** has an attribute named **date_time** which references a select type called **date_time_select**;
- in this case, the **date_time_select** references the **date_and_time** entity;
- the **date_and_time** entity has an attribute named **date_offset**;
- the attribute **date_offset** references the entity **date**.

– The application element **APPROVAL** has an attribute purpose which maps to the purpose attribute of the **approval** entity in the application interpreted model. The source of the attribute purpose in the entity **approval** is ISO 10303-41.

D.3 Examples to illustrate data exchange and data sharing

D.3.1 Data Exchange

An account statement detailing a customer's banking activity is prepared monthly by a bank and sent to a customer via the US Postal Service. The customer receives the statement and uses

Figure D.2 – Data exchange scenario

the information to balance his cheque book and track his financial position. A "data exchange" has taken place (see figure D.2).

Account information maintained by the bank has been transformed into a form the customer can understand (a statement), the information has been transported to the customer via an exchange mechanism (the US Postal Service), and the customer has received and understood the information. Such an exchange has the following characteristics:

– Initiated by data producer

Typically, a data exchange occurs upon the initiation of the party who originally produces the information, in this case the bank.

– Transformation to neutral format

The bank may choose among many different internal representations to effectively manage its account information. In order to communicate that information with the customer, however, a transformation must take place to convert the information into a form the customer can readily understand. In this scenario, the information to be exchanged includes not only the data (numbers representing dollar values), but also the context of that data. In this case, the context is represented by the column titles and descriptive text which accompany the numeric data on the statement. Without this context, the customer would have a difficult time understanding the information.

– Redundant copies of data

Once the exchange has taken place, there are now two copies of the account information:

Figure D.3 – Data sharing scenario

the account information represented internally at the bank and the statement now in the possession of the customer. Presumably the customer will use the account statement to maintain his own account information independent of the bank.

– Discrete event in time

A data exchange occurs as a discrete event in time. Once the exchange has been accomplished, the information at the "sending" side (the bank) may change without notice to the "receiving" side (the customer). The information passed during the exchange is simply a snapshot of the data as it existed at the time of the exchange. If this new information is to be passed along to the customer, another data exchange must take place. Similarly, if the customer wishes to respond in some way to the account statement, the customer must initiate a new data exchange in which his response is sent to the bank.

D.3.2 Data Sharing

Data sharing, unlike data exchange, is not easily defined to represent a commonly agreed upon technology. At an abstract level, data sharing means data stored in one system can be shared by other systems. Conceptually, a sharable data instance needs only be stored once in a single place and used for many purposes by many systems. The term "share" is to use, own, or receive jointly; it is also defined as a portion of a whole which is given or assigned. If we use these definitions, data sharing is not the act of transferring a snapshot of the data but to allow for joint use and ownership.

Consider one example (as illustrated in figure D.3) of data sharing in relation to the banking analogy presented above.

Rather than mailing an account statement every month, the bank may implement a touch-tone telephone system to pass along information to its customers. A customer may call the bank and use the push-buttons of his telephone to enter his account number and navigate menus to access and update his account information. In this case, data sharing has the following characteristics:

– Initiated by data receiver

Typically, data sharing occurs upon the initiation of the party who desires/receives the information, in this case the customer who initiates the telephone call.

– Data on demand

Because the telephone system provides the customer with the latest account information at the time of the phone call, the customer need only call at the time when the data is actually needed. In addition, the customer chooses the content of the information he wishes to receive by selectively navigating the touch tone menu system.

– Context embedded in protocol

The context of the information received by the customer is partly defined by the specific steps, or protocol, the customer uses in interacting with the phone system. For example, in response to the phone system's prompts, the customer may press "1" to receive chequing account information and then press "2" to receive current balance or "3" to receive most recent cheque number that has been cashed. The actual context of the final numeric data received by the customer is defined by the series of steps taken to receive the data.

Although the touch telephone banking system example helps to illustrate some of the characteristics of data sharing, there are additional characteristics which fall out of the scope of the imperfect example above:

– Single Data Instance

In most data sharing systems, only one instance, or copy, of the data is maintained. Other interested parties may access the information as needs require, but only one copy is to be considered the updated, legitimate repository of the data.

– Read and Update

Data sharing often encompasses more than just a retrieval of information. Facilities often exist which allow for the update of the single data repository by multiple parties. In this perspective, data sharing is a "two-way" street which allows for both information access as well as information update.

Annex E
(informative)

The generic product description resource

The generic product description resource is used within ISO 10303 to define a logical structure for product description and management data. This model is used as the integration framework for the ISO 10303integrated resources. The framework is designed to classify the nature of the product data into well defined concepts. These concepts are:

Application context: data that defines the purpose for which product information is created, and the types of product, disciplines, and life-cycle stages for which such information is valid. The use of an application context allows data that represents an "as designed" product to be distinguished from that for an "as built" configuration, etc.

Product identification: data that identifies products, including variants and categories, and that defines life-cycle "views" of products..

Product definition: the identification of a collection of characteristics of the product that forms a life-cycle or discipline view of the definition of the product. Product definition data also includes that which relates to the structure of products, in terms of assembly structures, configurations, effectivities, bills of materials, etc

Product property definition: data that characterises products by their properties, independent of the representation of properties.

EXAMPLE 34 – It is possible to identify the shape of an object, or aspects of the shape, as a property of the object, without providing a detailed description of shape using a CAD model, engineering drawing, etc.

Product property representation: data that formally describes the properties of a product, including multiple descriptions of the same property.

EXAMPLE 35 – The shape of an object may be identified, and then described in many different ways: a 3D CAD model, a physical mock-up, an engineering drawing, and a technical illustration make us of different representations of the same shape.

Product property presentation: data that defines the presentation of product information to support human communication.

EXAMPLE 36 – The shape of an object (the property) is represented by co-ordinate values, curves, surfaces, etc.; this representation is presented by assigning colours, line fonts, etc. and displaying the resulting picture on a workstation.

Annex F
(informative)

ISO 10303 document structure

ISO 10303 is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application protocols, application interpreted constructs, abstract test suites, implementation methods and conformance testing. These are described in ISO 10303-1. Figure F.1 shows the relationship between the elements of the ISO 10303 architecture described above, and the documentation of ISO 10303 as a standard. The elements of the architecture that are specific to an industrial application form the basis for application protocols: parts of ISO 10303 (200 series) that standardise the data specification for defined industry application need. Although abstract test suites are specified for each application protocol they are (for historical reasons) published separately as parts in the 300 series. The elements of the architecture that are shared between applications are standardised either as integrated resources (40 series and 100 series) or as application interpreted constructs (500 series).

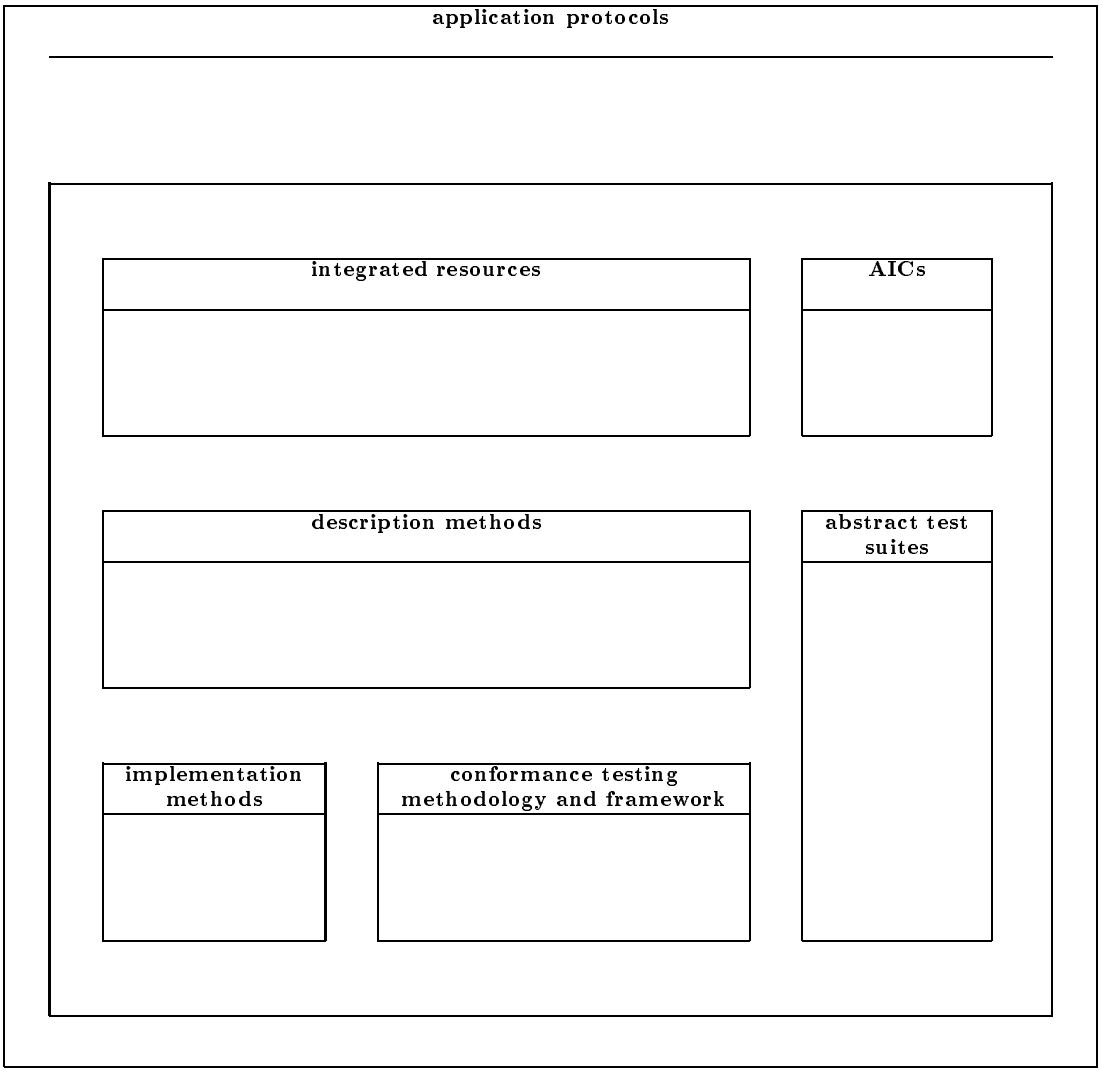

Figure F.1 – Relationship of the ISO 10303 architecture to the documentation of the standard

Annex G
(informative)

Functional aspects of the ISO 10303 architecture

The data architecture constitutes a conceptual framework that has four functional aspects that reflect the principles used in the development of ISO 10303. The functional aspects establish relationships among the architecture components. They are the data usage, data exchange, data specification and data integration aspects of the architecture.

G.1 Data usage aspect

The data usage aspect identifies a relationship between application domain knowledge about enterprise tasks, an application activity model, the scope of an industrial application, and an application reference model. In ISO 10303, application domain knowledge is provided by participants from industry and commerce that have first hand experience with the processes, knowledge, and participants used to accomplish enterprise tasks. The application activity model is a description of the activities used to accomplish such enterprise tasks. The scope of an industrial application describes those activities that are within the scope of an identified application for an application protocol. The application reference model is a description of application information used in carrying out application tasks. The fundamental principle of the data usage aspect is that the application activity model, the scope of an industrial application, and the application reference model are specified from an application domain perspective. They are principal elements of an application domain ontology established within an application protocol that describes the product data that exists from the perspective of its use in the specified industrial application that is the focus of the application protocol.

G.2 Data exchange aspect

The data exchange aspect identifies a relationship between implementation knowledge about the development of computer aided systems, the clear text encoding and standard data access interface (SDAI) implementation forms, and an application interpreted model. The clear text encoding is used to specify human readable text-based representation of data instances that populate an application interpreted model. The SDAI is used to specify access capabilities and computer language bindings that are used to access data instances that populate an application interpreted model. The application interpreted model is a description of application information that employs standard product data constructs from the integrated resources (integrated resources) and application interpreted constructs (application interpreted constructs). The fundamental principle of the data exchange aspect is that the application interpreted model is specified using standard constructs from a generic product data model, the integrated resources (integrated resources). The application interpreted model establishes a description of product data for a specific application using a data structure that represents basic product data semantics. The generic product data perspective accommodates the fact that any specific application describes only a portion of the product data about a given product. Other applications will cover different discipline specific perspectives and life-cycle phases that collectively define the totality

Figure G.1 – The usage, exchange, specification and integration aspects of the data architecture

of data available about a product. The integrated resources accommodate these concepts and provide the basis for data integration across application interpreted models.

G.3 Data specification aspect

The data specification aspect identifies a relationship between data modelling knowledge, the data usage aspect, and the data exchange aspect. The application reference model is the principal focus with respect to the data usage aspect. It is the data specification that describes the application domain perspective used in the data usage aspect. The application interpreted model is the principle focus with respect to the data exchange aspect. It is the data specification that describes a generic product data perspective used in the data exchange aspect. The relationship between the description of the information used in carrying out application tasks from a domain perspective and the description of that information from the perspective of generic product data semantics involves the specification of a transformation between the application reference model and the application interpreted model. This transformation is specified in mapping tables that describe how standard product data constructs of the integrated resources (integrated resources) are used in the development of the application interpreted model to represent the underlying semantics of the application reference model. Where identical semantics exist in two or more application interpreted models, application interpreted constructs are developed that are used by each application interpreted model. The fundamental principle of the data specification aspect is that there is a specified transformation between an application domain perspective and a generic product data perspective. The application domain perspective describes how enterprise participants think about the information they create and use. This description is valuable in
(1) validating the perspective within industry since it is the perspective used in accomplishing enterprise tasks,
(2) helping enterprise participants who wish to exchange product data to decide which application protocols deal with the information they use, and
(3) developing computer systems that reflect the way enterprise participants think about information while performing their tasks.

The generic product data perspective describes the application domain information in terms of a generic model of product data using standard data constructs. This description uses canonical product data constructs that are common to all product descriptions. Specification of the application interpreted model provides both the specificity necessary for effective product data exchange for an identified application and the generality necessary for accommodating information from many application perspectives that span many technical disciplines over multiple life-cycle phases of a product.

G.4 Data integration aspect

The data integration aspect identifies a relationship between data modelling knowledge (since it is a part of the data specification aspect), the integrated resources, the application interpreted constructs, and multiple application interpreted models from different application protocols (figure G.1). Each application interpreted model uses the canonical product data constructs of the integrated resources whether they are used directly from the integrated resources or

indirectly from the application interpreted constructs. These semantic resources are available to all application protocols in the development of a generic product data perspective for a given scope of an industrial application. An application activity model that has been developed to describe all discipline activities and life-cycle phases of a product also crosses application protocol boundaries and is useful in data integration.

if no conformance classes are defined for an application interpreted model, it is required that conforming implementations implement the complete application interpreted model, i.e., that the full application interpreted model is the sole conformance class defined.

The scope of an industrial application, application reference model, mapping tables, and application interpreted model are different for each application protocol though they may contain common elements. The application domain knowledge often crosses multiple application protocols. This varies among application domains. Data modelling knowledge and computer system implementation knowledge crosses application protocol boundaries and can be applied to all applications. Each source of knowledge is useful in providing insight into integration of data across application boundaries for a given product. The fundamental principle of the data integration aspect is the common use of canonical product data constructs across all application protocols that provide the ability to identify those elements of product data that cross application protocol boundaries and those that though unique to an application protocol, contribute to the totality of what is known about a product.